

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF: Mamoru SAKAMOTO

GAU:

SERIAL NO: New Application

EXAMINER:

FILED: Herewith

FOR: INTERRUPT CONTROLLING METHOD CAPABLE OF EXECUTING INTERRUPT PROCESS
WHILE AVOIDING SLOWING-DOWN OF EXECUTION SPEED OF TASK PROCESS

REQUEST FOR PRIORITY

COMMISSIONER FOR PATENTS
ALEXANDRIA, VIRGINIA 22313

SIR:

- ☐ Full benefit of the filing date of U.S. Application Serial Number , filed , is claimed pursuant to the provisions of 35 U.S.C. §120.
- ☐ Full benefit of the filing date(s) of U.S. Provisional Application(s) is claimed pursuant to the provisions of 35 U.S.C. §119(e):
Application No. Date Filed
- ☒ Applicants claim any right to priority from any earlier filed applications to which they may be entitled pursuant to the provisions of 35 U.S.C. §119, as noted below.

In the matter of the above-identified application for patent, notice is hereby given that the applicants claim as priority:

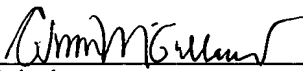
<u>COUNTRY</u>	<u>APPLICATION NUMBER</u>	<u>MONTH/DAY/YEAR</u>
Japan	2003-066292	March 12, 2003

Certified copies of the corresponding Convention Application(s)

- ☒ are submitted herewith
- ☐ will be submitted prior to payment of the Final Fee
- ☐ were filed in prior application Serial No. filed
- ☐ were submitted to the International Bureau in PCT Application Number
Receipt of the certified copies by the International Bureau in a timely manner under PCT Rule 17.1(a) has been acknowledged as evidenced by the attached PCT/IB/304.
- ☐ (A) Application Serial No.(s) were filed in prior application Serial No. filed ; and
- ☐ (B) Application Serial No.(s)
☐ are submitted herewith
☐ will be submitted prior to payment of the Final Fee

Respectfully Submitted,

OBLON, SPIVAK, McCLELLAND,
MAIER & NEUSTADT, P.C.


Marvin J. Spivak

Registration No. 24,913

C. Irvin McClelland
Registration Number 21,124

Customer Number

22850

Tel. (703) 413-3000
Fax. (703) 413-2220
(OSMMN 05/03)

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2003年 3月12日

出 願 番 号

Application Number:

特願2003-066292

[ST.10/C]:

[JP 2003-066292]

出 願 人

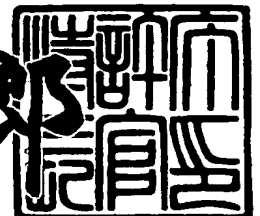
Applicant(s):

三菱電機株式会社

2003年 4月11日

特 許 庁 長 官
Commissioner,
Japan Patent Office

太田信一郎



出証番号 出証特2003-3025335

【書類名】 特許願

【整理番号】 542547JP01

【提出日】 平成15年 3月12日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/46

【発明者】

 【住所又は居所】 東京都千代田区丸の内二丁目2番3号 三菱電機株式会
社内

 【氏名】 坂本 守

【特許出願人】

 【識別番号】 000006013

 【氏名又は名称】 三菱電機株式会社

【代理人】

 【識別番号】 100089233

 【弁理士】

 【氏名又は名称】 吉田 茂明

【選任した代理人】

 【識別番号】 100088672

 【弁理士】

 【氏名又は名称】 吉竹 英俊

【選任した代理人】

 【識別番号】 100088845

 【弁理士】

 【氏名又は名称】 有田 貴弘

【手数料の表示】

 【予納台帳番号】 012852

 【納付金額】 21,000円

【提出物件の目録】

 【物件名】 明細書 1

【物件名】	図面	1
【物件名】	要約書	1
【プルーフの要否】	要	

【書類名】 明細書

【発明の名称】 割り込み制御方法

【特許請求の範囲】

【請求項 1】 (a) タスク処理を実行するための命令が記述された第 1 のプログラム中に、前記タスク処理を割り込み処理に対して排他的に実行することが必要な場合がある第 1 領域と、前記タスク処理を前記割り込み処理に対して排他的に実行することが必要でない第 2 領域とを規定するステップと、

(b) 前記第 1 のプログラムの実行中に割り込み要求が発生した場合、前記第 1 のプログラムの実行を中断し、前記第 2 領域中に、少なくとも一つのブレイクポイントを設定するステップと、

(c) 前記ステップ (b) で中断された前記第 1 のプログラムの実行を、前記ブレイクポイントの設定が完了した後に再開するステップと、

(d) 前記ステップ (c) で再開された前記第 1 のプログラムの実行が前記ブレイクポイントに到達した場合、前記第 1 のプログラムの実行を中断し、前記割り込み処理を実行するステップと、

(e) 前記割り込み処理の実行が完了した後に、前記ブレイクポイントの設定を解除するステップと
を備える、割り込み制御方法。

【請求項 2】 前記ブレイクポイントは複数であり、
前記第 2 領域内には、複数の分岐命令が含まれており、
前記ステップ (b) においては、前記第 2 領域の先頭番地、前記第 2 領域の最終番地、及び、前記複数の分岐命令が格納されている複数の番地に、前記ブレイクポイントがそれぞれ設定される、請求項 1 に記載の割り込み制御方法。

【請求項 3】 前記ブレイクポイントは複数であり、
前記第 2 領域内には、複数の分岐命令が含まれており、
前記ステップ (b) は、

(b-1) 前記第 1 のプログラムが中断された時点で実行中であった命令が格納されている番地を取得するステップと、

(b-2) 前記番地が前記第 1 領域に属する場合に実行され、前記第 2 領域の

先頭番地のみに前記ブレイクポイントを設定するステップと、

(b-3) 前記番地が前記第2領域に属する場合に実行され、前記第2領域の先頭番地、前記第2領域の最終番地、及び、前記複数の分岐命令が格納されている複数の番地に、前記ブレイクポイントをそれぞれ設定するステップとを有する、請求項1に記載の割り込み制御方法。

【請求項4】 (f) 割り込み処理許可フラグを設定するステップをさらに備え、

前記割り込み処理許可フラグが「許可」に設定されている場合において、前記第1のプログラムの実行中に前記割り込み要求が発生した場合には、前記第1のプログラムの実行を中断し、前記ステップ(b)～(e)を実行することなく、前記割り込み処理が実行され、

前記ステップ(b)～(e)は、前記割り込み処理許可フラグが「不許可」に設定されている場合に実行される、請求項1～3のいずれか一つに記載の割り込み制御方法。

【請求項5】 (a) タスク処理を実行するための命令が記述された第1のプログラム中に、前記タスク処理を割り込み処理に対して排他的に実行することが必要な場合がある第1領域と、前記タスク処理を前記割り込み処理に対して排他的に実行することが必要でない第2領域とを規定するステップと、

(b) 前記第1のプログラムの実行中に割り込み要求が発生した場合、前記第1のプログラムの実行を中断し、中断された時点で実行中であった命令が格納されている番地を取得するステップと、

(c) 前記番地が前記第1領域に属する場合には、前記第2領域中にブレイクポイントを設定し、一方、前記番地が前記第2領域に属する場合には、前記割り込み処理を実行するステップと、

(d) 前記ステップ(b)で中断された前記第1のプログラムの実行を、前記ステップ(c)の後に再開するステップと、

(e) 前記ステップ(c)で前記番地が前記第1領域に属していた場合において、前記ステップ(d)で再開された前記第1のプログラムの実行が前記ブレイクポイントに到達した場合、前記第1のプログラムの実行を中断し、前記割り込

み処理を実行するステップと、

(f) 前記ステップ(e)における前記割り込み処理の実行が完了した後に、前記ブレイクポイントの設定を解除するステップとを備える、割り込み制御方法。

【請求項6】 前記ステップ(c)において、前記ブレイクポイントは、前記第2領域の先頭番地のみに設定される、請求項5に記載の割り込み制御方法。

【請求項7】 (g) 割り込み処理許可フラグを設定するステップをさらに備え、

前記割り込み許可フラグが「許可」に設定されている場合において、前記第1のプログラムの実行中に前記割り込み要求が発生した場合には、前記第1のプログラムの実行を中断し、前記ステップ(b)～(f)を実行することなく、前記割り込み処理が実行され、

前記ステップ(b)～(f)は、前記割り込みフラグが「不許可」に設定されている場合に実行される、請求項5又は6に記載の割り込み制御方法。

【請求項8】 前記ブレイクポイントを設定するための処理は、

(x-1) 前記ブレイクポイントを設定しようとしている特定番地に格納されている第1の命令を保存するステップと、

(x-2) 前記割り込み処理を実行するための第2のプログラムの先頭番地への分岐を指示する第2の命令を、前記特定番地に書き込むステップとを有する、請求項1～7のいずれか一つに記載の割り込み制御方法。

【請求項9】 前記ブレイクポイントの設定を解除するための処理は、

(y) 前記特定番地に格納されている前記第2の命令を、前記第1の命令に書き換えるステップ

を有する、請求項8に記載の割り込み制御方法。

【請求項10】 前記ブレイクポイントを設定するための処理は、

(x-1) 前記ブレイクポイントを設定しようとしている特定番地を、所定のレジスタに設定するステップと、

(x-2) 前記第1のプログラムを実行する過程において、プログラムカウンタに設定されている番地と、前記所定のレジスタに設定されている前記特定番地

とを、逐次比較するステップと、

(x-3) 前記プログラムカウンタに設定されている前記番地と、前記所定のレジスタに設定されている前記特定番地とが一致した場合、前記割り込み処理を実行するための第2のプログラムの先頭番地を、前記プログラムカウンタに設定するステップと

を有する、請求項1～7のいずれか一つに記載の割り込み制御方法。

【請求項11】 前記ブレイクポイントの設定を解除するための処理は、

(y) 前記所定のレジスタの設定をクリアするステップ

を有する、請求項10に記載の割り込み制御方法。

【請求項12】 前記ブレイクポイントを設定するための処理は、

(x) シミュレータによって実行され、前記ブレイクポイントを設定しようとしている特定番地を、所定の変数に設定するステップ

を有する、請求項1～7のいずれか一つに記載の割り込み制御方法。

【請求項13】 前記ブレイクポイントを設定するための処理は、

(x) トランスレータによって実行され、前記ブレイクポイントを設定しようとしている特定番地を、所定の変数に設定するステップ

を有する、請求項1～7のいずれか一つに記載の割り込み制御方法。

【請求項14】 前記ブレイクポイントの設定を解除するための処理は、

(y) 前記所定の変数の設定をクリアするステップ

を有する、請求項12又は13に記載の割り込み制御方法。

【請求項15】 (a) タスク処理を実行するための命令が記述された第1のプログラム中に、所定の変数をレジスタにロードして使用する処理が実行される第1領域と、前記所定の変数を使用した処理が実行されない第2領域とを規定するステップと、

(b) 前記第1のプログラムの実行中に、前記所定の変数を使用した処理が実行される割り込み要求が発生した場合、前記第1のプログラムの実行を中断し、中断された時点で実行中であった命令が格納されている番地を取得するステップと、

(c) 前記番地が前記第2領域に属する場合において、割り込み処理を実行す

るステップと、

(d) 前記番地が前記第 1 領域に属する場合において、前記所定の変数を前記レジスタの内容に書き換え、書き換え後の前記所定の変数を用いて割り込み処理を実行するステップと
を備える、割り込み制御方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

この発明は、割り込み制御方法に関するものである。

【0002】

【従来の技術】

タスク処理と割り込み処理とを排他的に実行するために、従来の割り込み制御装置では、割り込みマスクを用いることにより、割り込みハンドラの起動の許可及び不許可を切り換える方式が採用されている。

【0003】

具体的には、割り込み処理に対して排他的に実行する必要がある第 1 のタスク処理の実行が開始される前に、割り込みマスクが「不許可」に設定される。そして、第 1 のタスク処理が完了した後、割り込み処理に対して排他的に実行する必要がない第 2 のタスク処理の実行が開始される前に、割り込みマスクが「許可」に設定される。

【0004】

なお、割り込みマスクを用いた割り込み制御装置に関する技術が、下記の特許文献 1 に開示されている。

【0005】

【特許文献 1】

特開 2 0 0 2 - 7 3 3 5 0 号公報

【0006】

【発明が解決しようとする課題】

しかしながら、このような従来の割り込み制御装置によると、割り込み要求が

発生していない場合であっても、本来のタスク処理以外に、割り込みマスクレジスタを設定するための処理が追加的に実行される。そのため、タスク処理の実行速度が低下してしまうという問題がある。

【 0 0 0 7 】

本発明はかかる問題を解決するために成されたものであり、タスク処理の実行速度が低下することを回避しつつ割り込み処理を実行することが可能な割り込み制御方法を得ることを目的とする。

【 0 0 0 8 】

【課題を解決するための手段】

第 1 の発明によれば、割り込み制御方法は、（a）タスク処理を実行するための命令が記述された第 1 のプログラム中に、タスク処理を割り込み処理に対して排他的に実行することが必要な場合がある第 1 領域と、タスク処理を割り込み処理に対して排他的に実行することが必要でない第 2 領域とを規定するステップと、（b）第 1 のプログラムの実行中に割り込み要求が発生した場合、第 1 のプログラムの実行を中断し、第 2 領域中に、少なくとも一つのブレイクポイントを設定するステップと、（c）ステップ（b）で中断された第 1 のプログラムの実行を、ブレイクポイントの設定が完了した後に再開するステップと、（d）ステップ（c）で再開された第 1 のプログラムの実行がブレイクポイントに到達した場合、第 1 のプログラムの実行を中断し、割り込み処理を実行するステップと、（e）割り込み処理の実行が完了した後に、ブレイクポイントの設定を解除するステップとを備える。

【 0 0 0 9 】

第 2 の発明によれば、割り込み制御方法は、（a）タスク処理を実行するための命令が記述された第 1 のプログラム中に、タスク処理を割り込み処理に対して排他的に実行することが必要な場合がある第 1 領域と、タスク処理を割り込み処理に対して排他的に実行することが必要でない第 2 領域とを規定するステップと、（b）第 1 のプログラムの実行中に割り込み要求が発生した場合、第 1 のプログラムの実行を中断し、中断された時点で実行中であった命令が格納されている番地を取得するステップと、（c）番地が第 1 領域に属する場合には、第 2 領域

中にブレイクポイントを設定し、一方、番地が第2領域に属する場合には、割り込み処理を実行するステップと、(d)ステップ(b)で中断された第1のプログラムの実行を、ステップ(c)の後に再開するステップと、(e)ステップ(c)で番地が第1領域に属していた場合において、ステップ(d)で再開された第1のプログラムの実行がブレイクポイントに到達した場合、第1のプログラムの実行を中断し、割り込み処理を実行するステップと、(f)ステップ(e)における割り込み処理の実行が完了した後に、ブレイクポイントの設定を解除するステップとを備える。

【0010】

【発明の実施の形態】

実施の形態1.

図1は、本発明の実施の形態1に係る割り込み制御装置の構成を模式的に示すブロック図である。割り込み制御装置は、マイクロプロセッサ1と、マイクロプロセッサ1によってアクセス可能な記憶部2とを備えている。記憶部2は、ROM、RAM等の半導体メモリである。

【0011】

図2は、記憶部2の記憶空間の一部を抜き出して示す模式図である。記憶部2には、タスク処理プログラムを構成する複数の命令、即ちタスク処理を実行するための複数の命令が格納されている。アドレスA1～Anには、割り込み処理に対して排他的に実行することが必要な複数の命令が格納されている。アドレスA1～Anに格納されている命令を実行することは、割り込み処理に対して排他的に実行することが必要な第1のタスク処理を実行することに等しい。また、アドレスB1～Bnには、割り込み処理に対して排他的に実行することが必要でない複数の命令が格納されている。アドレスB1～Bnに格納されている命令を実行することは、割り込み処理に対して排他的に実行することが必要でない第2のタスク処理を実行することに等しい。実際には、第1及び第2のタスク処理が繰り返し実行されることが多いが、説明の簡略化のため、本明細書では、第1及び第2のタスク処理が、それぞれ一回ずつ実行される場合について説明する。なお、図2において、アドレスB4、B7には、分岐命令がそれぞれ格納されているも

のとする。

【0012】

図3～6は、本実施の形態1に係る割り込み制御方法における処理の流れを示すフローチャートである。以下、図1～6を参照して、本実施の形態1に係る割り込み制御装置の動作について説明する。

【0013】

図3を参照して、まず、タスク処理プログラムの実行に先立って、開始時処理が一度だけ実行される。ステップSP1100において、マイクロプロセッサ1は、割り込み処理許可領域及び割り込み処理禁止領域を設定する。割り込み処理許可領域内に含めるべき命令及び割り込み処理禁止領域内に含めるべき命令は、例えば、タスク処理プログラムの設計者によって、プログラムの設計段階で適宜決定される。割り込み処理許可領域を特定する特定情報、及び割り込み処理禁止領域を特定する特定情報は、記憶部2のROM内に予め記憶されている。具体的に、ステップSP1100においてマイクロプロセッサ1は、ROM内に記憶されている上記2つの特定情報をロードして、記憶部2のRAM内に記憶する処理を行う。これによって、割り込み処理許可領域及び割り込み処理禁止領域が規定される。図4～6に示すその後の処理において、タスク処理プログラムを構成する各命令が割り込み処理許可領域中にあるか割り込み処理禁止領域中にあるかを判断する際には、記憶部2のRAMに記憶されている上記特定情報が参照されることになる。

【0014】

例えば図2を参照して、割り込み処理禁止領域を特定する特定情報は、その領域に属する命令のアドレスA1～Anで表現されており、割り込み処理許可領域を特定する特定情報は、その領域に属する命令のアドレスB1～Bnで表現されている。また、割り込み処理禁止領域及び割り込み処理許可領域の各々が、アドレスが連続する複数の命令で構成されている場合には、各特定情報は、割り込み処理禁止領域及び割り込み処理許可領域の各々の先頭アドレスと最終アドレスとによって表現してもよい。

【0015】

図4を参照して、次に、タスク処理プログラムの実行が開始される。ステップSP1200において、マイクロプロセッサ1は、割り込み処理に対して排他的に実行する必要があるタスク処理を実行する。具体的に、マイクロプロセッサ1は、図2のアドレスA1～Anに格納されている命令を順に実行する。次に、ステップSP1210において、マイクロプロセッサ1は、割り込み処理に対して排他的に実行する必要がないタスク処理を実行する。具体的に、マイクロプロセッサ1は、図2のアドレスB1～Bnに格納されている命令を順に実行する。

【0016】

マイクロプロセッサ1がタスク処理プログラムを実行している最中に、タイマの期限切れやシリアル通信デバイスからの送受信完了等に起因して、外部入力イベントがマイクロプロセッサ1に通知されたものとする。即ち、割り込み要求が発生したものとする。すると、マイクロプロセッサ1は、タスク処理プログラムの実行を中断して、図5に示す割り込みハンドラの実行を開始する。

【0017】

図5を参照して、まず、ステップSP1300において、マイクロプロセッサ1は、割り込みハンドラに移行する直前のマイクロプロセッサ1のコンテキスト（レジスタ値等）を保存する。次に、ステップSP1310において、マイクロプロセッサ1は、割り込み処理許可領域R2内に複数のブレイクポイントを設定する。ここで、ブレイクポイントとは、それが設定されたアドレスにプログラムの実行が到達した場合に、所定のプログラム（後述のブレイクポイントハンドラ）の実行が開始されるアドレスを意味する。本実施の形態1において、マイクロプロセッサ1は、割り込み処理許可領域R2の先頭アドレスB1、最終アドレスBn、及び、分岐命令が格納されている全てのアドレスB4、B7に、それぞれブレイクポイントを設定する。割り込み処理禁止領域R1内には、ブレイクポイントは設定されない。

【0018】

次に、ステップSP1320において、マイクロプロセッサ1は、ステップSP1300で保存しておいたコンテキストを復元する。次に、ステップSP1330において、マイクロプロセッサ1は、割り込まれたアドレスにジャンプする

。即ち、割り込みハンドラに移行する直前に実行されていた命令が格納されているアドレスにジャンプする。これにより、中断されていたタスク処理プログラムの実行が再開される。

【 0 0 1 9 】

図 4 に戻り、ステップ S P 1 2 1 0 において、タスク処理プログラムの実行は、図 5 のステップ S P 1 3 1 0 で設定された複数のブレイクポイントのいずれかに、やがて到達する。ここでは、ブレイクポイント B P 1 (図示しない) に到達したものとする。すると、マイクロプロセッサ 1 は、タスク処理プログラムの実行を中断し、図 6 に示すブレイクポイントハンドラの実行を開始する。ここで、ブレイクポイントハンドラは、タスク処理プログラムの実行がブレイクポイントに到達した場合に起動されるプログラムである。

【 0 0 2 0 】

図 6 を参照して、まず、ステップ S P 1 4 0 0 において、マイクロプロセッサ 1 は、ブレイクポイントハンドラに移行する直前のマイクロプロセッサ 1 のコンテキストを保存する。次に、ステップ S P 1 4 1 0 において、マイクロプロセッサ 1 は、タスク処理に対して排他的に実行する必要がある割り込み処理を実行する。次に、ステップ S P 1 4 2 0 において、マイクロプロセッサ 1 は、全てのブレイクポイントの設定を解除する。次に、ステップ S P 1 4 3 0 において、マイクロプロセッサ 1 は、ステップ S P 1 4 0 0 で保存しておいたコンテキストを復元する。次に、ステップ S P 1 4 4 0 において、マイクロプロセッサ 1 は、ブレイクポイント B P 1 のアドレスにジャンプする。これにより、中断されていたタスク処理プログラムの実行が再開される。

【 0 0 2 1 】

ブレイクポイントは割り込み処理禁止領域 R 1 内には設定されないため、割り込み処理禁止領域 R 1 内の命令を実行している最中に、ブレイクポイントハンドラが呼び出されることはない。これにより、割り込み処理に対して排他的に実行する必要があるタスク処理と、タスク処理に対して排他的に実行する必要がある割り込み処理との、排他的な実行が実現される。

【 0 0 2 2 】

このように本実施の形態 1 に係る割り込み制御方法及び割り込み制御装置によると、割り込み要求が発生していない通常動作中においては、排他制御のために本来のタスク処理に追加される処理（割り込みマスクの設定等）が全く行われな
い。従って、割り込み要求が発生しない限り、タスク処理には排他制御に起因するオーバーヘッドが無い
ため、タスク処理の実行速度が低下することを回避できる。

【 0 0 2 3 】

また、本実施の形態 1 に係る割り込み制御方法及び割り込み制御装置において、ブレイクポイントは、割り込み処理許可領域 R 2 の先頭アドレス B 1、最終アドレス B n、及び、分岐命令が格納されている全てのアドレス B 4、B 7 に、それぞれ設定される。ブレイクポイントの設定後にタスク処理プログラムが再開されてから、タスク処理プログラムの実行がいずれかのブレイクポイントに到達するまでの期間、割り込み処理の実行開始が遅延されることになる。しかしながら、一般的に、分岐命令は数命令に一回の頻度で出現することが多いため、遅延時間をごくわずかに抑えることができる。なお、遅延時間をさらに抑えるべく、分岐命令が格納されているアドレス以外にも、さらにブレイクポイントを設定してもよい。

【 0 0 2 4 】

実施の形態 2.

図 7 は、本発明の実施の形態 2 に係る割り込み制御方法における処理の流れを示すフローチャートである。本実施の形態 2 に係る割り込み制御装置は、図 5 に示した割り込みハンドラの処理の代わりに、図 7 に示す割り込みハンドラの処理を実行する。以下、図 1 ～ 4、6、7 を参照して、本実施の形態 2 に係る割り込み制御装置の動作について説明する。

【 0 0 2 5 】

図 3 を参照して、まず、上記実施の形態 1 と同様に、開始時処理が実行される。図 4 を参照して、次に、上記実施の形態 1 と同様に、タスク処理プログラムの実行が開始される。

【 0 0 2 6 】

マイクロプロセッサ1がタスク処理プログラムを実行している最中に、割り込み要求が発生したものとする。すると、マイクロプロセッサ1は、タスク処理プログラムの実行を中断して、図7に示す割り込みハンドラの実行を開始する。

【0027】

図7を参照して、まず、ステップSP2300において、マイクロプロセッサ1は、割り込みハンドラに移行する直前のマイクロプロセッサ1のコンテキストを保存する。次に、ステップSP2310において、マイクロプロセッサ1は、タスク処理プログラムが中断された時点で実行中であった命令が格納されているアドレスを取得する。次に、ステップSP2320において、マイクロプロセッサ1は、ステップSP2310で取得されたアドレスが、図2の割り込み処理禁止領域R1中にあるか否かを判定する。

【0028】

ステップSP2320における判定の結果が「YES」である場合、即ち、ステップSP2310で取得されたアドレスが割り込み処理禁止領域R1中にある場合は、ステップSP2330に進み、マイクロプロセッサ1は、図2の割り込み処理許可領域R2の先頭アドレスB1のみにブレイクポイントを設定する。

【0029】

一方、ステップSP2320における判定の結果が「NO」である場合、即ち、ステップSP2310で取得されたアドレスが割り込み処理許可領域R2中にある場合は、ステップSP2340に進み、マイクロプロセッサ1は、ブレイクポイントの設定処理を実行することなく、割り込み要求に対応する割り込み処理を直ちに実行する。

【0030】

ステップSP2330、SP2340に引き続き、ステップSP2350において、マイクロプロセッサ1は、ステップSP2300で保存しておいたコンテキストを復元する。次に、ステップSP2360において、マイクロプロセッサ1は、割り込まれたアドレスへジャンプする。これにより、中断されていたタスク処理プログラムの実行が再開される。

【0031】

ステップ S P 2 3 2 0 における判定の結果が「Y E S」であった場合、図 4 を参照して、再開されたタスク処理プログラムの実行は、割り込み処理許可領域 R 2 の先頭アドレス B 1 に設定されたブレイクポイントに、やがて到達する。すると、上記実施の形態 1 と同様に、図 6 に示したブレイクポイントハンドラによって割り込み処理が実行される。

【 0 0 3 2 】

このように本実施の形態 2 に係る割り込み制御方法及び割り込み制御装置によると、上記実施の形態 1 と同様に、通常動作中においては、本来のタスク処理に追加される処理が全く行われな。従って、タスク処理の実行速度が低下することを回避できる。

【 0 0 3 3 】

また、ステップ S P 2 3 1 0 で取得されたアドレスが割り込み処理許可領域 R 2 中にある場合、マイクロプロセッサ 1 は、割り込み処理を直ちに実行する。従って、割り込み処理の実行開始が遅延することを回避できる。

【 0 0 3 4 】

さらに、ステップ S P 2 3 1 0 で取得されたアドレスが割り込み処理禁止領域 R 1 中にある場合、マイクロプロセッサ 1 は、割り込み処理許可領域 R 2 の先頭アドレス B 1 のみにブレイクポイントを設定する。従って、上記実施の形態 1 と比較すると、ブレイクポイントを設定するための処理を簡略化することができる。

【 0 0 3 5 】

実施の形態 3 .

図 8 は、本発明の実施の形態 3 に係る割り込み制御方法における処理の流れを示すフローチャートである。本実施の形態 3 に係る割り込み制御装置は、図 5 に示した割り込みハンドラの処理の代わりに、図 8 に示す割り込みハンドラの処理を実行する。以下、図 1 ～ 4 , 6 , 8 を参照して、本実施の形態 3 に係る割り込み制御装置の動作について説明する。

【 0 0 3 6 】

図 3 を参照して、まず、上記実施の形態 1 と同様に、開始時処理が実行される

。図4を参照して、次に、上記実施の形態1と同様に、タスク処理プログラムの実行が開始される。

【0037】

マイクロプロセッサ1がタスク処理プログラムを実行している最中に、割り込み要求が発生したものとする。すると、マイクロプロセッサ1は、タスク処理プログラムの実行を中断して、図8に示す割り込みハンドラの実行を開始する。

【0038】

図8を参照して、図7に示したステップSP2300～SP2320の処理と同様に、ステップSP3300～SP3320の処理が実行される。

【0039】

ステップSP3320における判定の結果が「YES」である場合は、ステップSP3330に進み、図7に示したステップSP2330と同様に、マイクロプロセッサ1は、割り込み処理許可領域R2の先頭アドレスB1のみにブレイクポイントを設定する。

【0040】

一方、ステップSP3320における判定の結果が「NO」である場合は、ステップSP3340に進み、図5に示したステップSP1310と同様に、マイクロプロセッサ1は、割り込み処理許可領域R2の先頭アドレスB1、最終アドレスBn、及び、分岐命令が格納されている全てのアドレスB4、B7に、それぞれブレイクポイントを設定する。

【0041】

ステップSP3330、SP3340に引き続き、図7に示したステップSP2350、SP2360の処理と同様に、ステップSP3350、SP3360の処理が実行される。

【0042】

ステップSP3320における判定の結果が「YES」であった場合、図4を参照して、再開されたタスク処理プログラムの実行は、割り込み処理許可領域R2の先頭アドレスB1に設定されたブレイクポイントに、やがて到達する。すると、上記実施の形態1と同様に、図6に示したブレイクポイントハンドラによっ

て割り込み処理が実行される。

【0043】

ステップSP3320における判定の結果が「NO」であった場合、図4を参照して、再開されたタスク処理プログラムの実行は、割り込み処理許可領域R2の先頭アドレスB1、最終アドレスBn、及び、分岐命令が格納されている全てのアドレスB4、B7にそれぞれ設定された複数のブレイクポイントのいずれかに、やがて到達する。すると、上記実施の形態1と同様に、図6に示したブレイクポイントハンドラによって割り込み処理が実行される。

【0044】

このように本実施の形態3に係る割り込み制御方法及び割り込み制御装置によると、ステップSP3310で取得されたアドレスが割り込み処理禁止領域R1中にある場合、マイクロプロセッサ1は、割り込み処理許可領域R2の先頭アドレスB1のみにブレイクポイントを設定する。従って、上記実施の形態1と比較すると、ブレイクポイントを設定するための処理を簡略化することができる。

【0045】

実施の形態4.

上記実施の形態1～3では、割り込み処理許可領域R2中に一又は複数のブレイクポイントが設定された。本実施の形態4では、ブレイクポイントを設定するための、及びブレイクポイントの設定を解除するための、第1の例について説明する。なお、本実施の形態4において、図1に示した記憶部2は、読み出し及び書き込みが可能なメモリ（RAM）である。

【0046】

図9は、ブレイクポイントを設定するための第1の例を示すフローチャートである。ステップSP4500において、マイクロプロセッサ1は、ブレイクポイントを設定しようとしているアドレス（以下「特定アドレス」と称する）に格納されている命令を、記憶部2内の所定の領域（以下「特定領域」と称する）に保存する。次に、ステップSP4510において、マイクロプロセッサ1は、ブレイクポイントハンドラの先頭アドレスへの分岐命令（ソフトウェアトラップ命令でも良い）を、特定アドレスに書き込む。

【 0 0 4 7 】

図 1 0 は、ブレイクポイントの設定を解除するための第 1 の例を示すフローチャートである。ステップ S P 4 6 0 0 において、マイクロプロセッサ 1 は、特定アドレスに格納されている、ブレイクポイントハンドラの先頭アドレスへの分岐命令を、ステップ S P 4 5 0 0 で特定領域に保存しておいた命令に書き換える。

【 0 0 4 8 】

このように本実施の形態 4 に係るブレイクポイントの設定方法によると、タスク処理プログラムの実行がブレイクポイントに到達すると、ブレイクポイントハンドラの先頭アドレスへ分岐されるため、ブレイクポイントハンドラの実行を確実に開始することができる。

【 0 0 4 9 】

また、本実施の形態 4 に係るブレイクポイントの設定解除方法によると、ブレイクポイントハンドラの先頭アドレスへの分岐命令が、元の命令に書き換えられるため、ブレイクポイントの設定が解除された後にブレイクポイントハンドラの実行が開始されることを回避することができる。

【 0 0 5 0 】

実施の形態 5.

本実施の形態 5 では、ブレイクポイントを設定するための、及びブレイクポイントの設定を解除するための、第 2 の例について説明する。

【 0 0 5 1 】

図 1 1 は、本実施の形態 5 に係る割り込み制御装置の構成を模式的に示すブロック図である。マイクロプロセッサ 1 は、プログラムカウンタ 3 と、レジスタ 4₁ ~ 4_n とを有している。記憶部 2 は、少なくとも読み出しが可能なメモリ（ROM, RAM）である。

【 0 0 5 2 】

図 1 2 は、ブレイクポイントを設定するための第 2 の例を示すフローチャートである。ステップ S P 5 5 0 0 において、マイクロプロセッサ 1 は、ブレイクポイントを設定しようとしている特定アドレスを、レジスタ 4₁ ~ 4_n に設定する。準備されているレジスタ 4₁ ~ 4_n の個数だけ、異なる特定アドレスを設定するこ

とができる。

【0053】

図13は、タスク処理プログラムが実行されている場合にマイクロプロセッサ1によって行われる処理の流れを示すフローチャートである。ステップSP5700において、マイクロプロセッサ1は、プログラムカウンタ3に設定されているアドレスと、レジスタ4₁～4_nに設定されている特定アドレスとが一致するかどうかを、逐次判定する。

【0054】

ステップSP5700における判定の結果が「NO」である場合は、ステップSP5710に進み、マイクロプロセッサ1は、プログラムカウンタ3に設定されているアドレスに格納されている命令を実行する。その命令の実行が完了すると、プログラムカウンタ3の設定値が更新される。プログラムカウンタ3に設定されているアドレスが、レジスタ4₁～4_nに設定されている特定アドレスのいずれかに一致するまで、ステップSP5700、SP5710の処理が繰り返される。即ち、タスク処理プログラムが実行される。

【0055】

ステップSP5700における判定の結果が「YES」である場合、即ち、プログラムカウンタ3に設定されているアドレスと、レジスタ4₁～4_nに設定されている特定アドレスのいずれかとが一致した場合は、ステップSP5720に進み、ブレイクポイントハンドラの先頭アドレスが、プログラムカウンタ3に設定される。これにより、ステップSP5720に続くステップSP5710において、ブレイクポイントハンドラの実行が開始される。

【0056】

図14は、ブレイクポイントの設定を解除するための第2の例を示すフローチャートである。ステップSP5600において、マイクロプロセッサ1は、全てのレジスタ4₁～4_nの値を-1に設定することにより、レジスタ4₁～4_nの設定をクリアする。

【0057】

このように本実施の形態5に係るブレイクポイントの設定方法によると、プロ

グラムカウンタ3に設定されているアドレスと、レジスタ $4_1 \sim 4_n$ に設定されている特定アドレスのいずれかとが一致すると、ブレイクポイントハンドラの先頭アドレスがプログラムカウンタ3に設定される。これにより、ブレイクポイントハンドラの実行を確実に開始することができる。

【0058】

しかも、記憶部2はROMで足りるため、上記実施の形態4と比較すると、記憶部2に関してコストの低減を図ることができる。

【0059】

また、本実施の形態5に係るブレイクポイントの設定解除方法によると、レジスタ $4_1 \sim 4_n$ の設定がクリアされるため、ブレイクポイントの設定が解除された後にブレイクポイントハンドラの実行が開始されることを回避することができる。

【0060】

実施の形態6.

本実施の形態6では、ブレイクポイントを設定するための、及びブレイクポイントの設定を解除するための、第3の例について説明する。本実施の形態6において、マイクロプロセッサ1は、シミュレータを実行する機能を有している。シミュレータは、シミュレーションを行うためのプログラムである。なお、シミュレーション機能を有するシステムでは、シミュレータを実行するマイクロプロセッサと、シミュレーションの対象となるマイクロプロセッサとが異なるのが一般的であるが、本実施の形態6では、両マイクロプロセッサはいずれもマイクロプロセッサ1である。

【0061】

図15は、タスク処理の流れを示すフローチャートである。上記実施の形態1と同様に、ステップSP6200において、マイクロプロセッサ1は、割り込み処理に対して排他的に実行する必要があるタスク処理を、タスク処理プログラムによって実行する。次に、ステップSP6210において、マイクロプロセッサ1は、割り込み処理に対して排他的に実行する必要がないタスク処理を、シミュレータによって実行する。

【0062】

図16は、ブレイクポイントを設定するための第3の例を示すフローチャートである。ステップSP6500において、マイクロプロセッサ1は、ブレイクポイントを設定しようとしている特定アドレスを、所定の変数 $C_1 \sim C_n$ に設定する。変数 $C_1 \sim C_n$ の個数だけ、異なる特定アドレスを設定することができる。

【0063】

図17は、図15のステップSP6210の処理内容を具体的に示すフローチャートである。ステップSP6700において、SPC (Simulation Program Counter) 変数が、割り込み処理許可領域R2の先頭アドレスB1に設定される。SPC変数は、シミュレータで実行中の命令が格納されているアドレスを保持する変数である。次に、ステップSP6710において、マイクロプロセッサ1は、SPC変数に設定されているアドレスが、割り込み処理許可領域R2の最終アドレスBnに一致するか否かを判定する。

【0064】

ステップSP6710における判定の結果が「YES」である場合は、処理が終了される。一方、ステップSP6710における判定の結果が「NO」である場合は、ステップSP6720に進み、マイクロプロセッサ1は、SPC変数に設定されているアドレスと、変数 $C_1 \sim C_n$ に設定されている特定アドレスのいずれかと一致するか否かを、逐次判定する。

【0065】

ステップSP6720における判定の結果が「NO」である場合は、ステップSP6730に進み、マイクロプロセッサ1は、SPC変数に設定されているアドレスに格納されている命令を実行する。その命令の実行が完了すると、SPC変数の設定値が更新される。その後、ステップSP6710における判定が再び実行される。

【0066】

ステップSP6720における判定の結果が「YES」である場合、即ち、SPC変数に設定されているアドレスと、変数 $C_1 \sim C_n$ に設定されている特定アドレスのいずれかと一致する場合は、マイクロプロセッサ1は、シミュレータの

実行を中断する。そして、図 6 に示した手順に従い、割り込み処理を実行し（ステップ S P 6 7 4 0）、ブレイクポイントの設定を解除する（ステップ S P 6 7 5 0）。その後、シミュレータの実行に戻り、ステップ S P 6 7 1 0 における判定が再び実行される。

【 0 0 6 7 】

図 1 8 は、ブレイクポイントの設定を解除するための第 3 の例を示すフローチャートであり、図 1 7 のステップ S P 6 7 5 0 の処理内容を具体的に示したものに相当する。ステップ S P 6 6 0 0 において、マイクロプロセッサ 1 は、全ての変数 $C_1 \sim C_n$ の値を - 1 に設定することにより、変数 $C_1 \sim C_n$ の設定をクリアする。

【 0 0 6 8 】

上記実施の形態 5 に係るブレイクポイントの設定方法では、設定できるブレイクポイントの個数の上限は、マイクロプロセッサ 1 が有するレジスタ $4_1 \sim 4_n$ の個数に制限される。従って、多数のブレイクポイントを設定したい場合には、それに応じて多数のレジスタ $4_1 \sim 4_n$ を準備する必要があり、コストが上昇する。これに対し、本実施の形態 6 に係るブレイクポイントの設定方法によると、変数 $C_1 \sim C_n$ を保存しておくためのメモリ容量の範囲内で、ブレイクポイントを設定できる。そのため、多数のブレイクポイントを設定したい場合であっても、コストの上昇を抑制することができる。

【 0 0 6 9 】

また、本実施の形態 6 に係るブレイクポイントの設定解除方法によると、変数 $C_1 \sim C_n$ の設定がクリアされるため、ブレイクポイントの設定が解除された後にブレイクポイントハンドラの実行が開始されることを回避することができる。

【 0 0 7 0 】

実施の形態 7.

本実施の形態 7 では、ブレイクポイントを設定するための、及びブレイクポイントの設定を解除するための、第 4 の例について説明する。本実施の形態 7 において、マイクロプロセッサ 1 は、トランスレータを実行する機能を有している。トランスレータは、トランスレーション（変換）を行うためのプログラムである

。トランスレーション実行では、ある命令が別の命令に変換され、変換後の命令が実行される。なお、トランスレーション機能を有するシステムでは、トランスレータを実行するマイクロプロセッサと、変換後の命令を実行するマイクロプロセッサとが異なるのが一般的であるが、本実施の形態 7 では、両マイクロプロセッサはいずれもマイクロプロセッサ 1 である。

【 0 0 7 1 】

図 1 9 は、タスク処理の流れを示すフローチャートである。上記実施の形態 1 と同様に、ステップ S P 7 2 0 0 において、マイクロプロセッサ 1 は、割り込み処理に対して排他的に実行する必要があるタスク処理を、タスク処理プログラムによって実行する。次に、ステップ S P 7 2 1 0 において、マイクロプロセッサ 1 は、割り込み処理に対して排他的に実行する必要がないタスク処理を、トランスレータによって実行する。

【 0 0 7 2 】

図 2 0 は、図 1 9 のステップ S P 7 2 1 0 の処理内容を具体的に示すフローチャートである。ステップ S P 7 7 0 0 において、T P C (Translation Program Counter) 変数が、割り込み処理許可領域 R 2 の先頭アドレス B 1 に設定される。T P C 変数は、トランスレータで実行中のプログラムを構成する複数の命令のうち、そのとき変換しようとしている命令が格納されているアドレスを保持する変数である。次に、ステップ S P 7 7 1 0 において、マイクロプロセッサ 1 は、T P C 変数に設定されているアドレスが、割り込み処理許可領域 R 2 の最終アドレス B n に一致するまで、トランスレーション実行を継続する。

【 0 0 7 3 】

本実施の形態 7 に係る、ブレイクポイントを設定するための第 4 の例では、上記実施の形態 6 と同様に、マイクロプロセッサ 1 は、ブレイクポイントを設定しようとしている特定アドレスを、所定の変数 $C_1 \sim C_n$ に設定する。また、本実施の形態 7 に係る、ブレイクポイントの設定を解除するための第 4 の例では、上記実施の形態 6 と同様に、マイクロプロセッサ 1 は、全ての変数 $C_1 \sim C_n$ の値を - 1 に設定することにより、変数 $C_1 \sim C_n$ の設定をクリアする。

【 0 0 7 4 】

図21, 22は、図20のステップSP7710の処理内容を具体的に示すフローチャートである。ステップSP7800において、マイクロプロセッサ1は、TPC変数に設定されているアドレスが、割り込み処理許可領域R2の最終アドレスB_nに一致しているか否かを判定する。

【0075】

ステップSP7800における判定の結果が「YES」である場合は、処理が終了される。一方、ステップSP7800における判定の結果が「NO」である場合は、ステップSP7810に進み、マイクロプロセッサ1は、変換後の命令を格納するためのバッファを空にする。次に、ステップSP7820において、マイクロプロセッサ1は、TPC変数に設定されているアドレスと、変数C₁～C_nに設定されている特定アドレスのいずれかとが一致するか否かを、逐次判定する。

【0076】

ステップSP7820における判定の結果が「YES」である場合、即ち、TPC変数に設定されているアドレスと、変数C₁～C_nに設定されている特定アドレスのいずれかとが一致した場合は、マイクロプロセッサ1は、ブレイクポイントハンドラをサブルーチンとして実行するようなネイティブ命令を生成し、そのネイティブ命令をバッファ中に追加する。

【0077】

ステップSP7830に引き続き、及び、ステップSP7820における判定の結果が「NO」である場合は、ステップSP7840において、マイクロプロセッサ1は、TPC変数に設定されているアドレスに格納されている命令が、無条件分岐命令であるか否かを判定する。

【0078】

ステップSP7840における判定の結果が「YES」である場合は、ステップSP7900に進み、マイクロプロセッサ1は、分岐先のアドレスからトランスレーション実行するようなネイティブ命令を生成し、そのネイティブ命令をバッファの最後に追加する。このネイティブ命令は、トランスレーション処理の先頭へ分岐するようなネイティブ命令である。一方、ステップSP7840におけ

る判定の結果が「NO」である場合は、ステップSP7850に進み、マイクロプロセッサ1は、TPC変数に設定されているアドレスに格納されている命令が、条件付き分岐命令であるか否かを判定する。

【0079】

ステップSP7850における判定の結果が「YES」である場合は、ステップSP7880に進み、マイクロプロセッサ1は、その条件が満足されているときは分岐先のアドレスからトランスレーション実行し、その条件が満足されていないときはTPC変数に設定されているアドレスの次のアドレスからトランスレーション実行するようなネイティブ命令を生成し、そのネイティブ命令をバッファ中に追加する。

【0080】

ステップSP7900、SP7880に引き続き、ステップSP7890において、バッファの先頭へのジャンプが実行される。その結果、これまでにバッファ中に蓄積されたネイティブ命令が、トランスレーション実行されることになる。このとき、ステップSP7830で生成されたネイティブ命令（即ちブレイクポイントハンドラを実行するようなネイティブ命令）がバッファ中に存在していれば、そのネイティブ命令がトランスレーション実行されることにより、ブレイクポイントハンドラが実行される。即ち、割り込み処理が実行される。さらに、バッファの最後には、トランスレーション処理に戻る命令が格納されているので、バッファ中に蓄積されているネイティブ命令のトランスレーション実行が完了した後は、トランスレーション処理（図21の先頭）に戻り、以降、実行が継続されることになる。

【0081】

ステップSP7850における判定の結果が「NO」である場合は、ステップSP7860に進み、マイクロプロセッサ1は、TPC変数に設定されているアドレスに格納されている命令に対応するネイティブ命令を生成し、そのネイティブ命令をバッファの最後に追加する。次に、ステップSP7870において、次の命令が格納されているアドレスが、TPC変数に設定される。即ち、TPC変数が更新される。その後、ステップSP7820における判定が再び実行される

【0082】

本実施の形態7に係るブレイクポイントの設定方法及びブレイクポイントの設定解除方法によっても、上記実施の形態6と同様の効果を得ることができる。

【0083】

実施の形態8.

図23, 24は、上記実施の形態1を基礎として、本実施の形態8に係る割り込み制御方法における処理の流れを示すフローチャートである。本実施の形態8に係る割り込み制御装置は、図4に示したタスク処理プログラムの処理の代わりに、図23に示すタスク処理プログラムの処理を実行する。また、図5に示した割り込みハンドラの処理の代わりに、図24に示す割り込みハンドラの処理を実行する。

【0084】

図23を参照して、まず、ステップSP8200において、割り込み処理許可フラグが「許可」又は「不許可」に設定される。次に、ステップSP8210において、マイクロプロセッサ1は、割り込み処理に対する排他的実行が必要な場合があるタスク処理を実行する。次に、ステップSP8220において、マイクロプロセッサ1は、割り込み処理に対して排他的に実行する必要がないタスク処理を実行する。

【0085】

図24を参照して、ステップSP8300において、マイクロプロセッサ1は、割り込みハンドラに移行する直前のマイクロプロセッサ1のコンテキストを保存する。次に、ステップSP8310において、マイクロプロセッサ1は、割り込み処理許可フラグが「許可」に設定されているか否かを判定する。

【0086】

ステップSP8310における判定の結果が「YES」である場合、即ち割り込み処理許可フラグが「許可」に設定されている場合は、ステップSP8320に進み、マイクロプロセッサ1は、ブレイクポイントの設定処理を行うことなく、割り込み処理を直ちに実行する。

【0087】

一方、ステップSP8310における判定の結果が「NO」である場合、即ち割り込み処理許可フラグが「不許可」に設定されている場合は、ステップSP8330に進み、マイクロプロセッサ1は、図5のステップSP1310と同様に、割り込み処理許可領域R2内に複数のブレイクポイントを設定する。

【0088】

ステップSP8320, SP8330に引き続き、ステップSP8340において、マイクロプロセッサ1は、ステップSP8300で保存しておいたコンテキストを復元する。次に、ステップSP8350において、マイクロプロセッサ1は、割り込まれたアドレスへジャンプする。これにより、中断されていたタスク処理プログラムの実行が再開される。

【0089】

ステップSP8310における判定の結果が「NO」であった場合、再開されたタスク処理プログラムの実行は、ステップSP8330で設定された複数のブレイクポイントのいずれかに、やがて到達する。すると、上記実施の形態1と同様に、図6に示したブレイクポイントハンドラによって割り込み処理が実行される。

【0090】

なお、以上の説明では、上記実施の形態1を基礎として本実施の形態8に係る発明を適用する例について述べたが、上記実施の形態2～7を基礎として、本実施の形態8に係る発明を適用することも可能である。

【0091】

このように本実施の形態8に係る割り込み制御方法及び割り込み制御装置によれば、タスク処理と割り込み処理とを排他的に実行するか非排他的に実行するかを、割り込み処理許可フラグの設定によって切り換えることができる。

【0092】

実施の形態9.

タスク処理と割り込み処理とを排他的に実行しなければならない理由は、以下の通りである。即ち、タスク処理及び割り込み処理の両者が共通にアクセスする

変数（グローバル変数）が存在し、一方がそれを変更し、他方がそれを参照するという状況において、一方によるグローバル変数の変更が完了する前に、他方によってグローバル変数を使用した処理が行われることを、回避しなければならないからである。

【 0 0 9 3 】

例えば、タスク処理によって所定のグローバル変数Gが変更され、割り込み処理によってグローバル変数Gが参照される状況を考える。このとき、タスク処理においては、グローバル変数Gを所定のレジスタにロードする第1ステップと、レジスタの設定値を用いて所定の処理を実行した後、レジスタの設定値を変更する第2ステップと、グローバル変数Gを、変更後のレジスタの設定値に書き換える第3ステップとが、この順に実行される。ここで、第3ステップが完了する前にグローバル変数Gを用いて割り込み処理を実行してしまうと、変更前のグローバル変数Gが参照されることとなり、支障が発生する。

【 0 0 9 4 】

本実施の形態9では、かかる支障の発生を適切に回避し得る、割り込み制御方法及び割り込み制御装置について説明する。なお、説明の簡略化のため、グローバル変数Gが1個のみ存在する場合を例にとり説明する。

【 0 0 9 5 】

図25は、本実施の形態9に係る割り込み制御装置の構成を模式的に示すブロック図である。マイクロプロセッサ1は、所定のレジスタ6を有している。

【 0 0 9 6 】

図26は、記憶部2の記憶空間の一部を抜き出して示す模式図である。記憶部2には、タスク処理プログラムを構成する複数の命令、即ちタスク処理を実行するための複数の命令が格納されている。アドレスA1には、グローバル変数Gをレジスタ6へロードすることを指示する命令が格納されている。アドレスA8には、グローバル変数Gをレジスタ6の設定値に書き換えることを指示する命令が格納されている。アドレスA2～A7に格納された命令によって、レジスタ6の設定値を用いた処理が実行された後、レジスタ6の設定値が変更される。アドレスA9～Anに格納された命令によっては、グローバル変数Gを用いたタスク処

理が実行されない。即ち、アドレスA9～Anに格納された命令を実行することによるタスク処理は、本来、割り込み処理に対して排他的に実行する必要がないタスク処理である。

【0097】

図27～30は、本実施の形態9に係る割り込み制御方法における処理の流れを示すフローチャートである。以下、図25～30を参照して、本実施の形態9に係る割り込み制御装置の動作について説明する。

【0098】

図27を参照して、まず、タスク処理プログラムの実行に先立って、開始時処理が一度だけ実行される。ステップSP9100において、マイクロプロセッサ1は、グローバル変数使用領域及びグローバル変数不使用領域を設定する。グローバル変数使用領域内に含めるべき命令及びグローバル変数不使用領域内に含めるべき命令は、例えば、タスク処理プログラムの設計者によって、プログラムの設計段階で適宜決定される。グローバル変数使用領域を特定する特定情報、及びグローバル変数不使用領域を特定する特定情報は、記憶部2のROM内に予め記憶されている。具体的に、ステップSP9100においてマイクロプロセッサ1は、ROM内に記憶されている上記2つの特定情報をロードして、記憶部2のRAM内に記憶する処理を行う。これによって、グローバル変数使用領域及びグローバル変数不使用領域が設定される。図28～30に示すその後の処理において、タスク処理プログラムを構成する各命令がグローバル変数使用領域中にあるかグローバル変数不使用領域中にあるかを判断する際には、記憶部2のRAMに記憶されている上記特定情報が参照されることになる。

【0099】

例えば図26を参照して、グローバル変数使用領域R1aを特定する特定情報は、その領域に属する命令のアドレスA1～A8で表現されており、グローバル変数不使用領域R1bを特定する特定情報は、その領域に属する命令のアドレスA9～Anで表現されている。また、グローバル変数使用領域及びグローバル変数不使用領域の各々が、アドレスが連続する複数の命令で構成されている場合には、各特定情報は、グローバル変数使用領域及びグローバル変数不使用領域の各

々の先頭アドレスと最終アドレスとによって表現してもよい。

【0100】

次に、ステップSP9110において、マイクロプロセッサ1は、図28に示す所定のルーチンを登録する。図28を参照して、ステップSP9400では、グローバル変数Gがレジスタ6の設定値に書き換えられる。ステップSP9410では、変更後のグローバル変数Gを用いて割り込み処理が実行される。ステップSP9420では、グローバル変数Gがレジスタ6にロードされる。

【0101】

図29を参照して、開始時処理が完了した後、タスク処理プログラムの実行が開始される。ステップSP9200において、マイクロプロセッサ1は、グローバル変数Gをレジスタ6にロードする。次に、ステップSP9210において、マイクロプロセッサ1は、レジスタ6の設定値を使用した所定のタスク処理を実行する。このタスク処理によって、レジスタ6の設定値が変更される。次に、ステップSP9220において、マイクロプロセッサ1は、グローバル変数Gを、変更後のレジスタ6の設定値に書き換える。次に、ステップSP9230において、マイクロプロセッサ1は、割り込み処理に対して排他的に実行する必要がないタスク処理を実行する。

【0102】

マイクロプロセッサ1がタスク処理プログラムを実行している最中に、割り込み要求が発生したものとする。すると、マイクロプロセッサ1は、タスク処理プログラムの実行を中断して、図30に示す割り込みハンドラの実行を開始する。

【0103】

図30を参照して、まず、ステップSP9300において、マイクロプロセッサ1は、割り込みハンドラに移行する直前のマイクロプロセッサ1のコンテキストを保存する。次に、ステップSP9310において、マイクロプロセッサ1は、割り込みハンドラが起動された時点（即ち、タスク処理プログラムが中断された時点）で実行中であった命令が格納されているアドレスを取得する。次に、ステップSP9320において、マイクロプロセッサ1は、ステップSP9310で取得されたアドレスが、図26のグローバル変数使用領域R1a中にあるか否

かを判定する。

【0104】

ステップSP9320における判定の結果が「NO」である場合、即ち、ステップSP9310で取得されたアドレスがグローバル変数不使用領域R1b中にある場合は、ステップSP9330に進み、マイクロプロセッサ1は、割り込み処理を実行する。

【0105】

一方、ステップSP9320における判定の結果が「YES」である場合、即ち、ステップSP9310で取得されたアドレスがグローバル変数使用領域R1a中にある場合は、ステップSP9340に進み、マイクロプロセッサ1は、図28に示したルーチンを呼び出して実行する。

【0106】

ステップSP9330、SP9340に引き続き、図7に示したステップSP2350、SP2360の処理と同様に、ステップSP9350、SP9360の処理が実行される。

【0107】

このように本実施の形態9に係る割り込み制御方法及び割り込み制御装置によると、割り込み要求が発生していない通常動作中においては、本来のタスク処理に追加される処理（割り込みマスクの設定等）が全く行われず、従って、割り込み要求が発生しない限り、タスク処理には排他制御に起因するオーバーヘッドが無い場合、タスク処理の実行速度が低下することを回避できる。

【0108】

また、グローバル変数使用領域R1a中のいずれかの命令が実行されている最中に割り込み要求が発生すると、予め登録しておいた所定のルーチンが呼び出される。そして、そのルーチンにおいては、割り込み処理の実行が開始される前に、グローバル変数Gがレジスタ6の設定値に書き換えられる。従って、割り込み処理がグローバル変数Gを参照する時には、グローバル変数Gは、タスク処理の実行によって更新された最新の値にすでに書き換えられている。その結果、更新前のグローバル変数Gが割り込み処理において参照されてしまうことに起因する

上記支障の発生を、適切に回避することができる。

【0109】

【発明の効果】

第1及び第2の発明によると、割り込み要求が発生しない限り、タスク処理には排他制御に起因するオーバーヘッドが無いため、タスク処理の実行速度が低下することを回避できる。

【図面の簡単な説明】

【図1】 本発明の実施の形態1に係る割り込み制御装置の構成を模式的に示すブロック図である。

【図2】 記憶部の記憶空間の一部を抜き出して示す模式図である。

【図3】 本発明の実施の形態1に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図4】 本発明の実施の形態1に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図5】 本発明の実施の形態1に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図6】 本発明の実施の形態1に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図7】 本発明の実施の形態2に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図8】 本発明の実施の形態3に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図9】 ブレイクポイントを設定するための第1の例を示すフローチャートである。

【図10】 ブレイクポイントの設定を解除するための第1の例を示すフローチャートである。

【図11】 本発明の実施の形態5に係る割り込み制御装置の構成を模式的に示すブロック図である。

【図12】 ブレイクポイントを設定するための第2の例を示すフローチャ

ートである。

【図 1 3】 タスク処理プログラムが実行されている場合にマイクロプロセッサによって行われる処理の流れを示すフローチャートである。

【図 1 4】 ブレイクポイントの設定を解除するための第 2 の例を示すフローチャートである。

【図 1 5】 タスク処理の流れを示すフローチャートである。

【図 1 6】 ブレイクポイントを設定するための第 3 の例を示すフローチャートである。

【図 1 7】 図 1 5 のステップ S P 6 2 1 0 の処理内容を具体的に示すフローチャートである。

【図 1 8】 ブレイクポイントの設定を解除するための第 3 の例を示すフローチャートである。

【図 1 9】 タスク処理の流れを示すフローチャートである。

【図 2 0】 図 1 9 のステップ S P 7 2 1 0 の処理内容を具体的に示すフローチャートである。

【図 2 1】 図 2 0 のステップ S P 7 7 1 0 の処理内容を具体的に示すフローチャートである。

【図 2 2】 図 2 0 のステップ S P 7 7 1 0 の処理内容を具体的に示すフローチャートである。

【図 2 3】 本発明の実施の形態 8 に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図 2 4】 本発明の実施の形態 8 に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図 2 5】 本発明の実施の形態 9 に係る割り込み制御装置の構成を模式的に示すブロック図である。

【図 2 6】 記憶部の記憶空間の一部を抜き出して示す模式図である。

【図 2 7】 本発明の実施の形態 9 に係る割り込み制御方法における処理の流れを示すフローチャートである。

【図 2 8】 本発明の実施の形態 9 に係る割り込み制御方法における処理の

流れを示すフローチャートである。

【図 2 9】 本発明の実施の形態 9 に係る割り込み制御方法における処理の流れを示すフローチャートである。

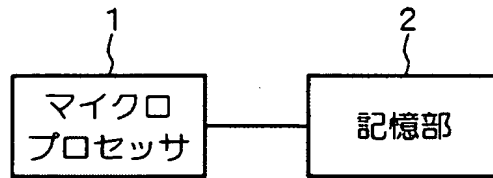
【図 3 0】 本発明の実施の形態 9 に係る割り込み制御方法における処理の流れを示すフローチャートである。

【符号の説明】

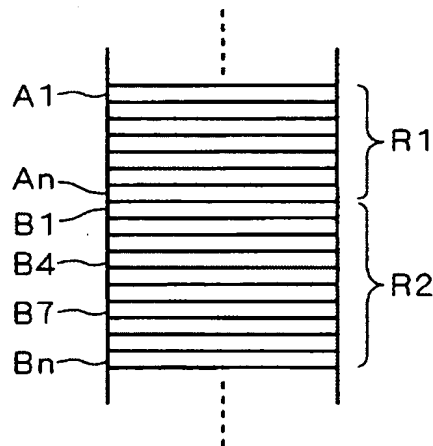
1 マイクロプロセッサ、2 記憶部、3 プログラムカウンタ、 $4_1 \sim 4_n$ 、
6 レジスタ。

【書類名】 図面

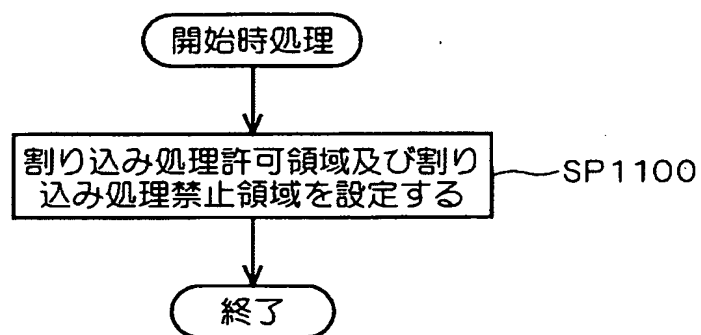
【図 1】



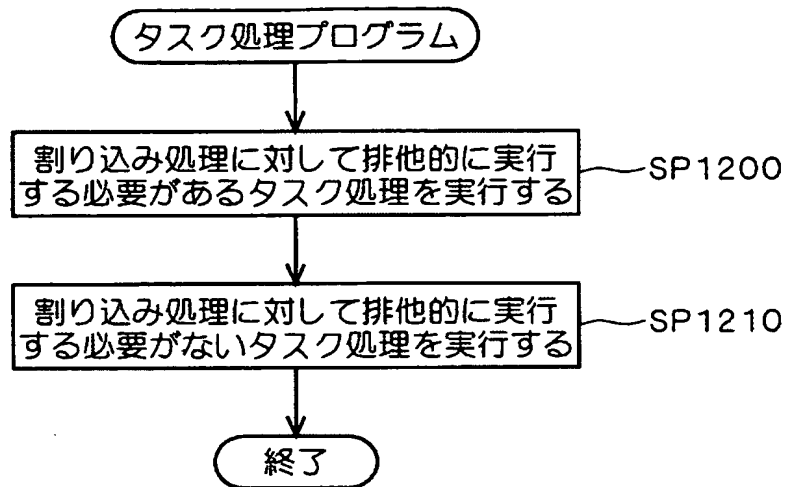
【図 2】



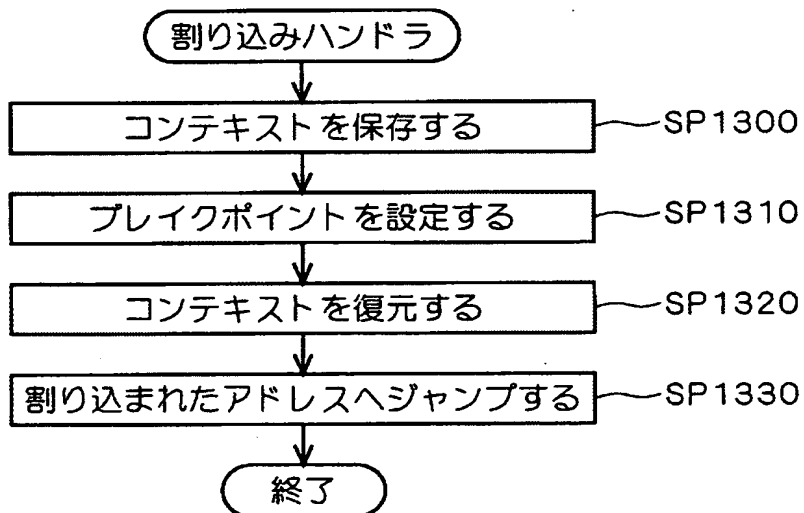
【図 3】



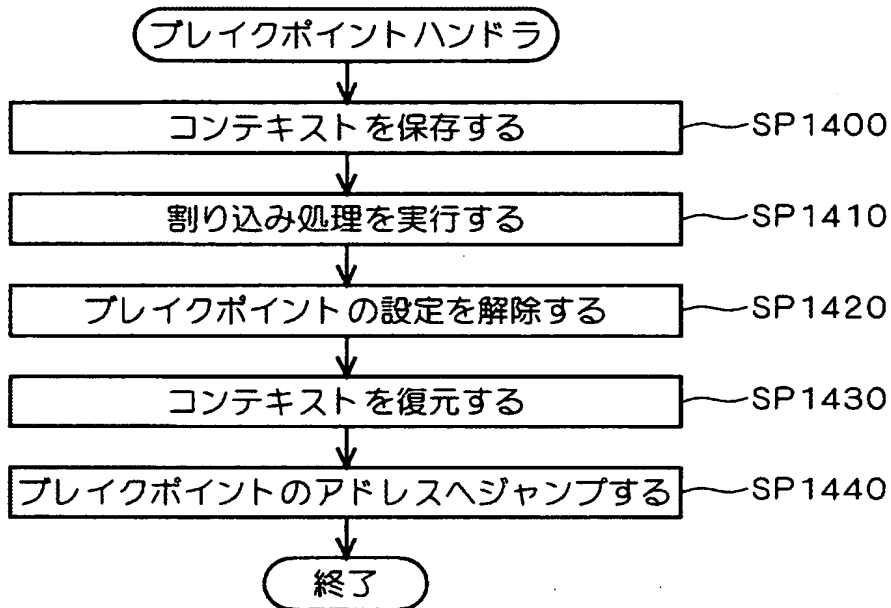
【図 4】



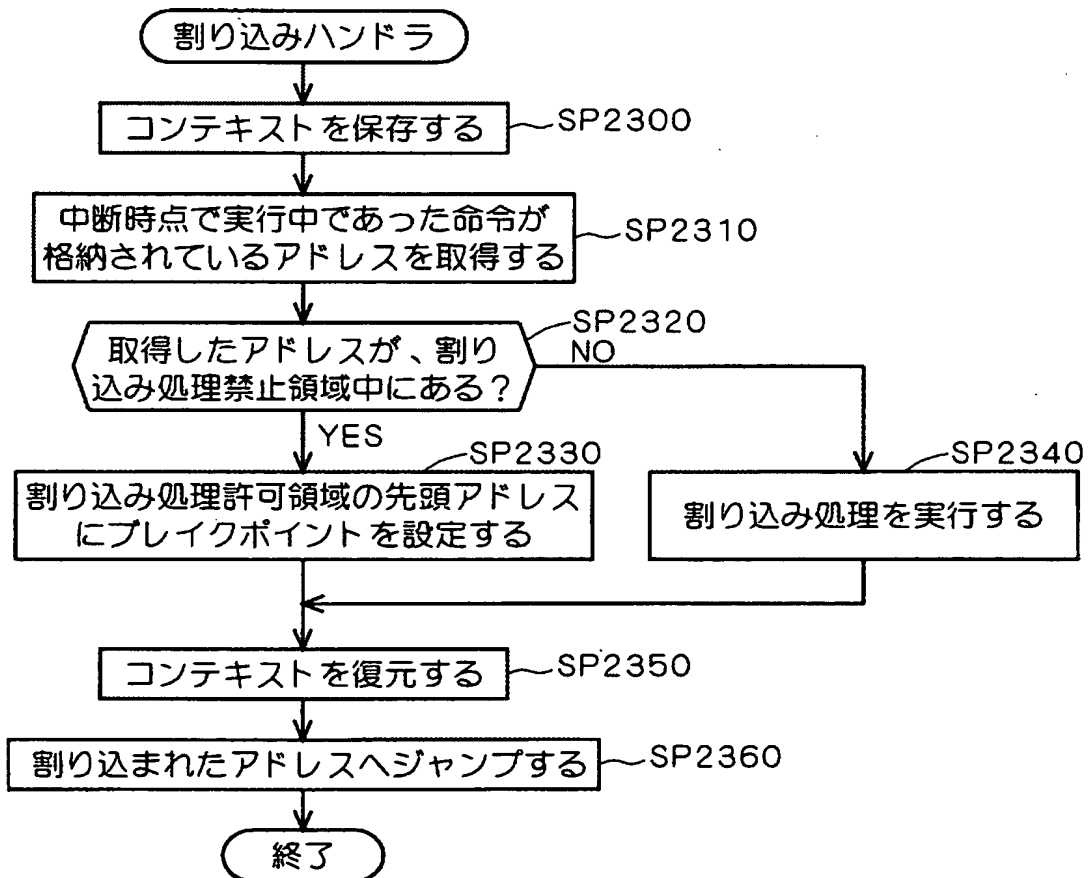
【図 5】



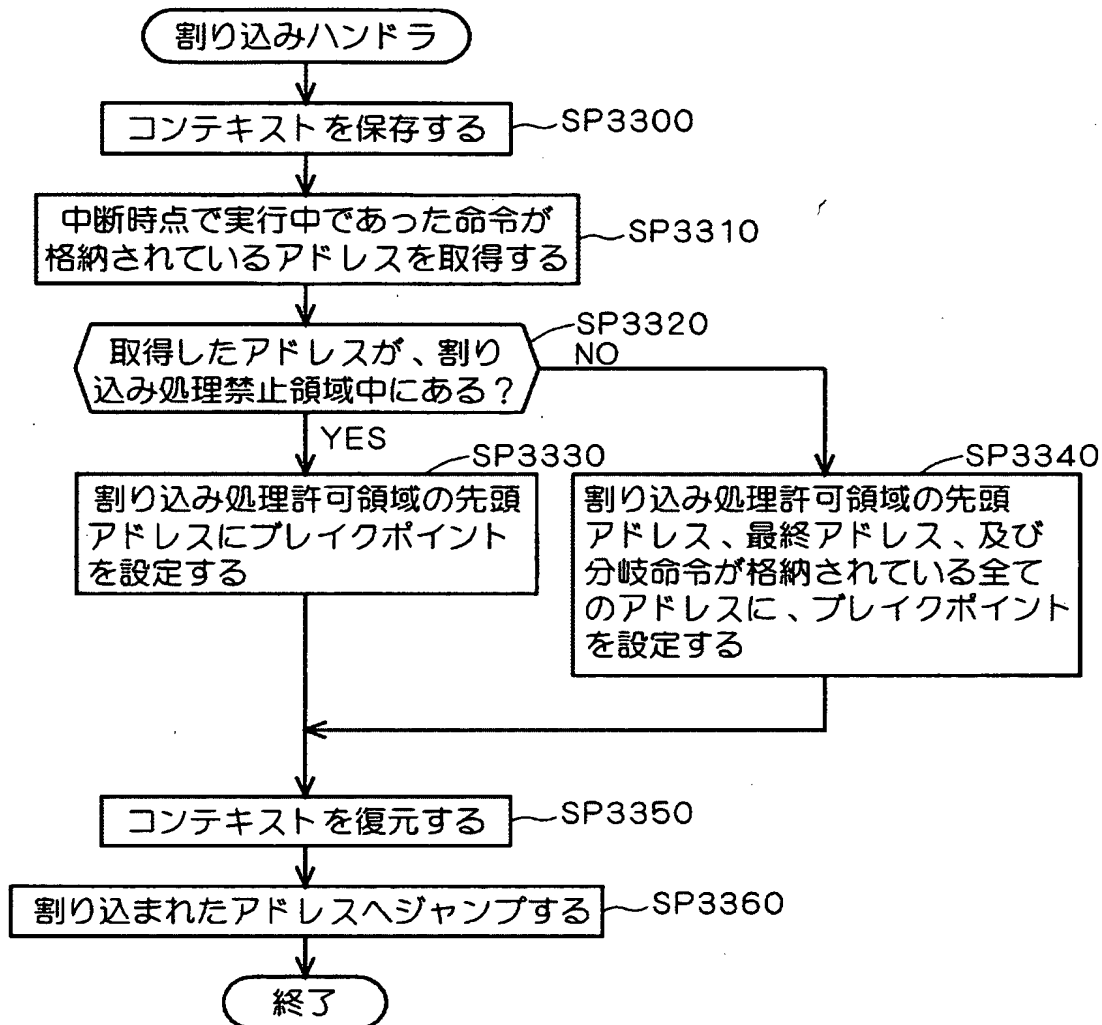
【図 6】



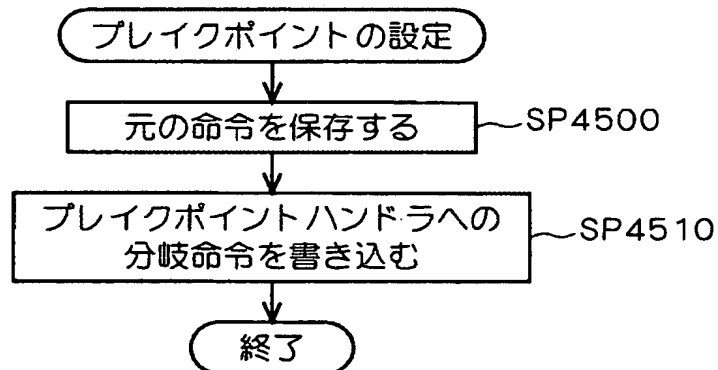
【図 7】



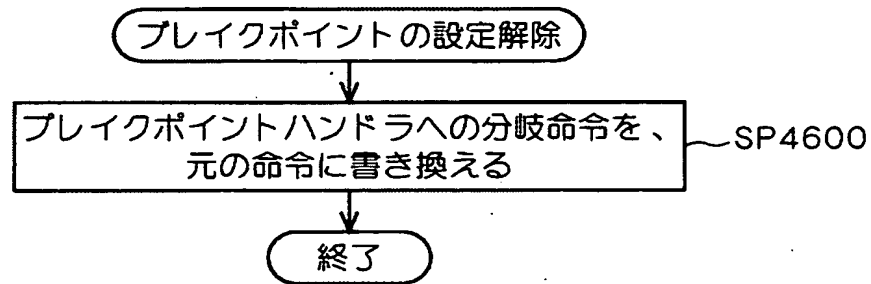
【図 8】



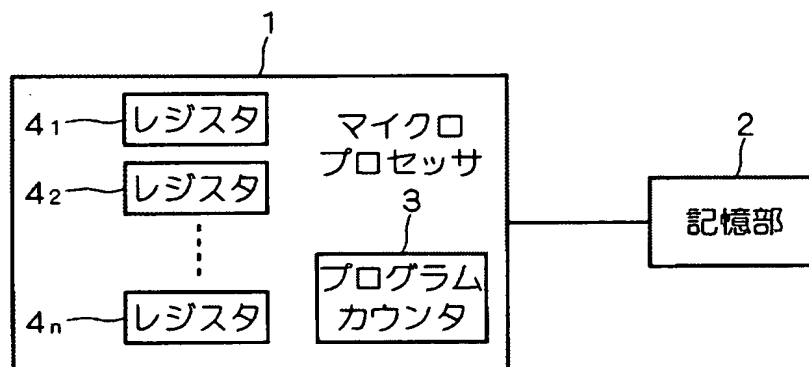
【図 9】



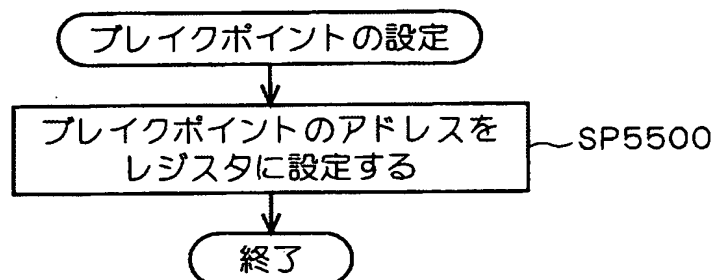
【図 10】



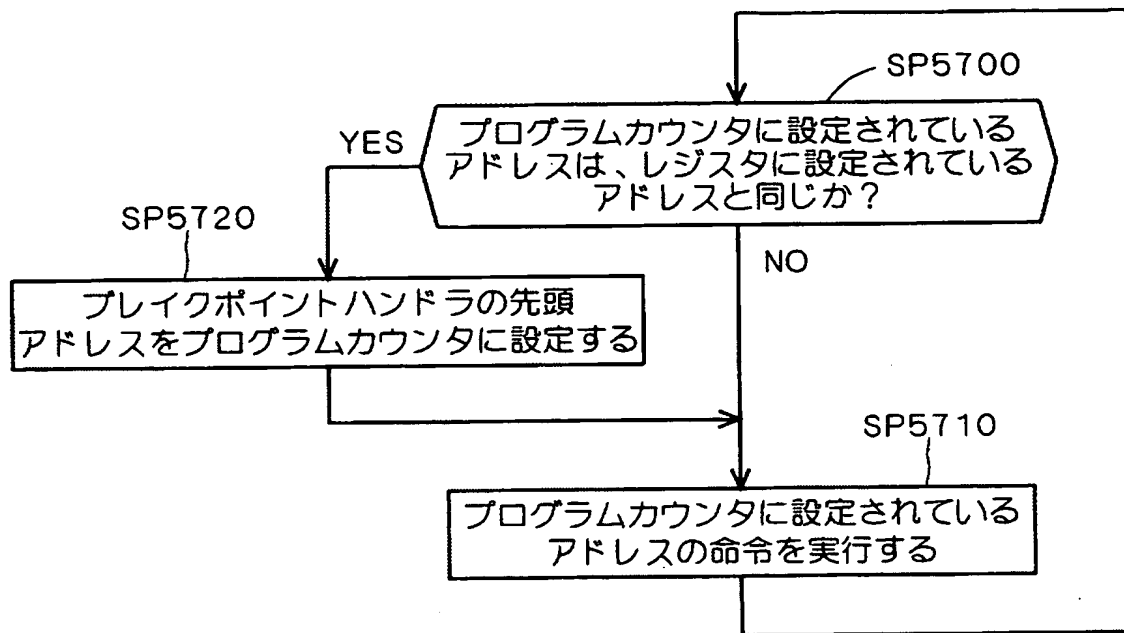
【図 11】



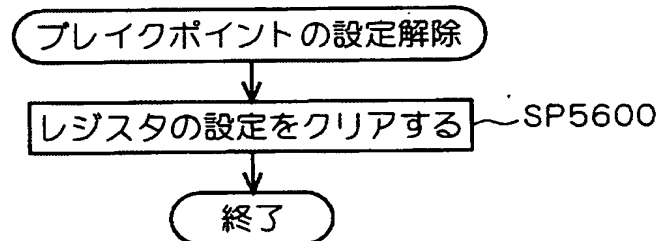
【図 12】



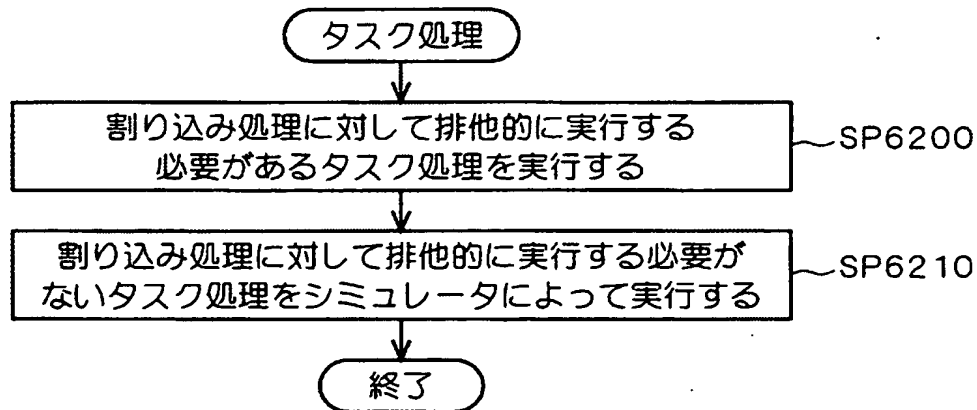
【図 13】



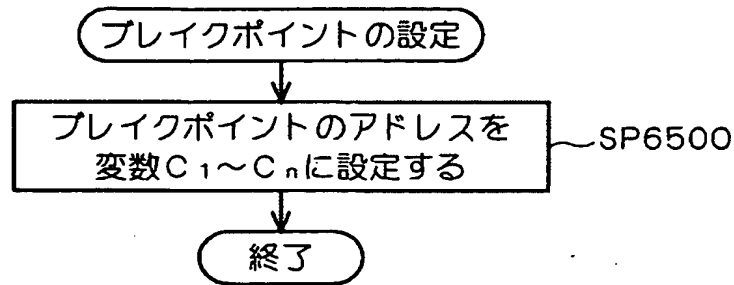
【図 14】



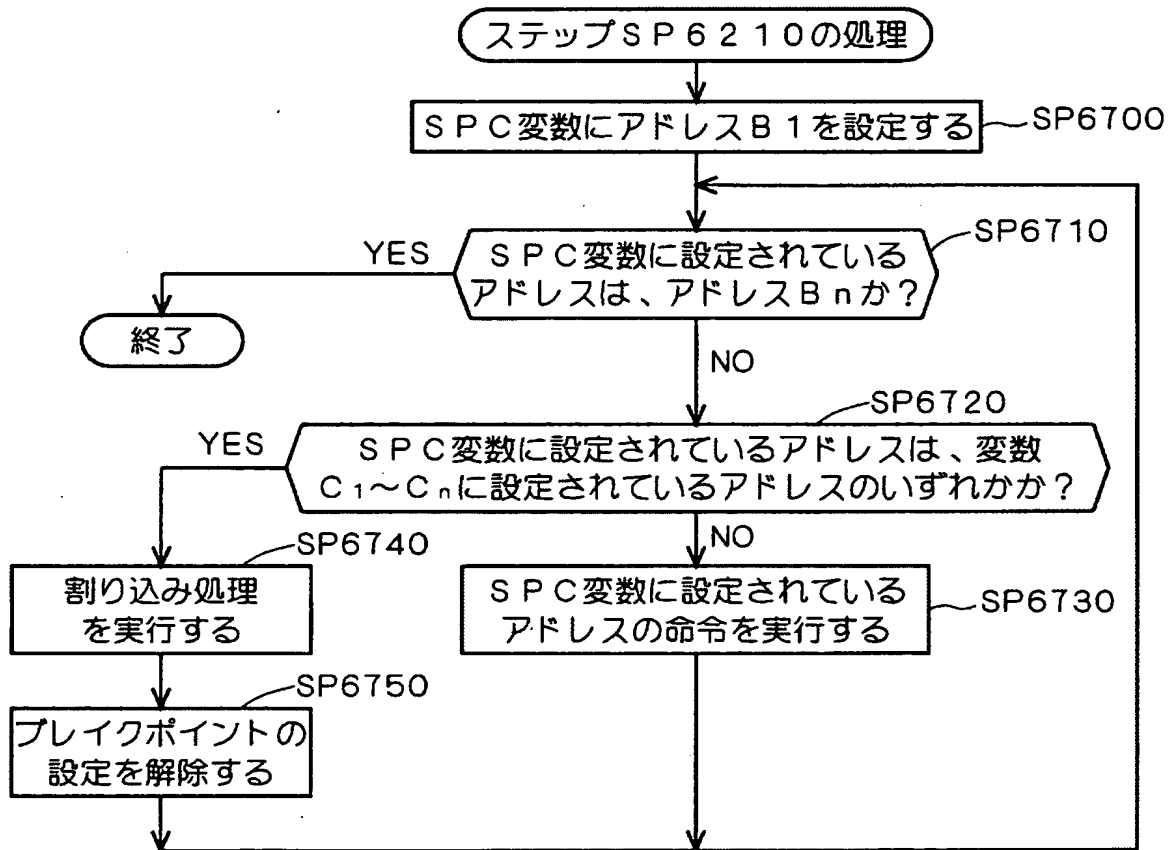
【図 15】



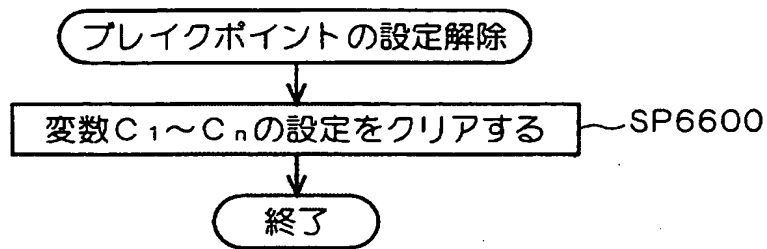
【図 16】



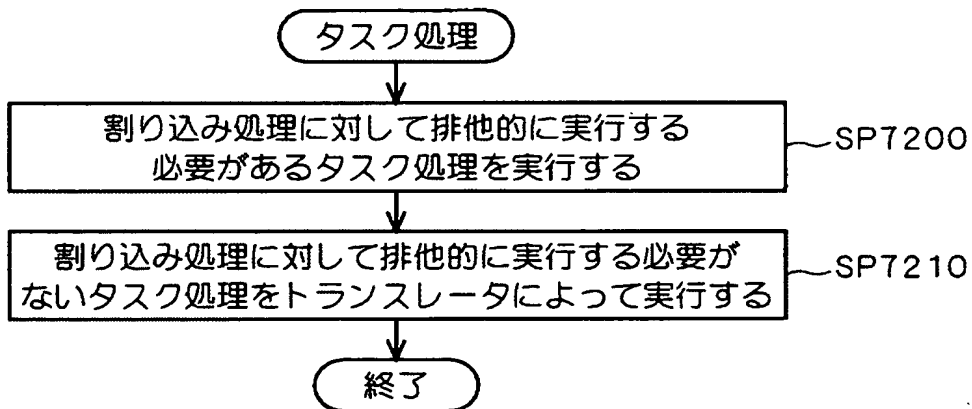
【図 17】



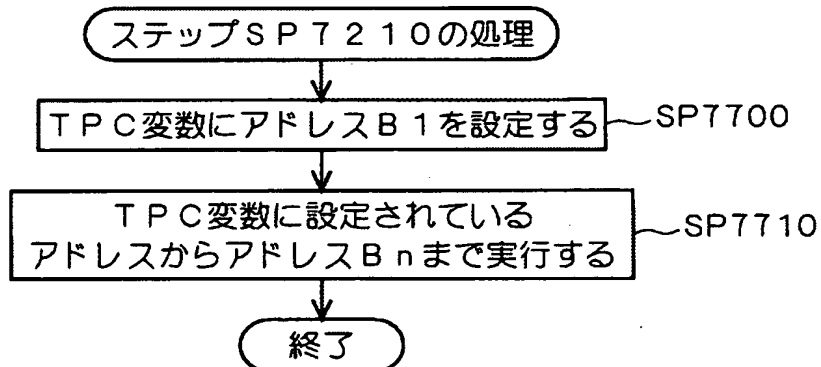
【図 1 8】



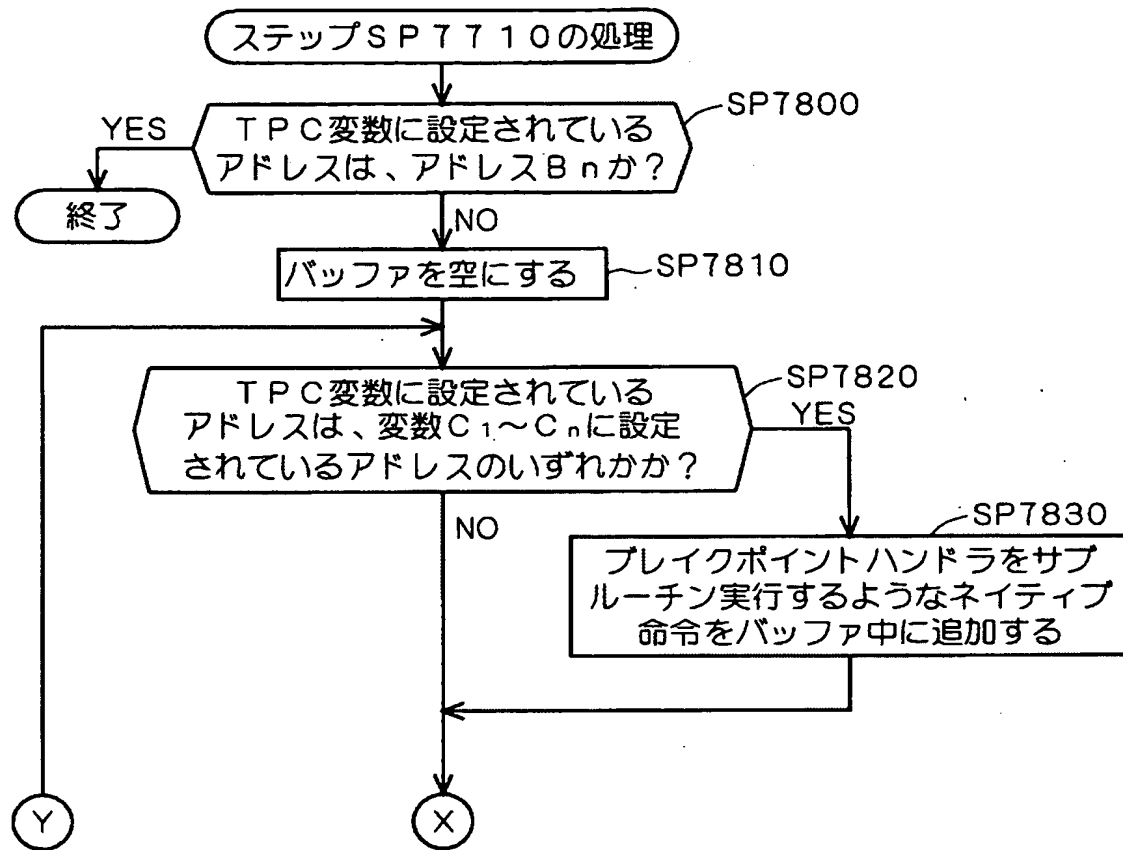
【図 1 9】



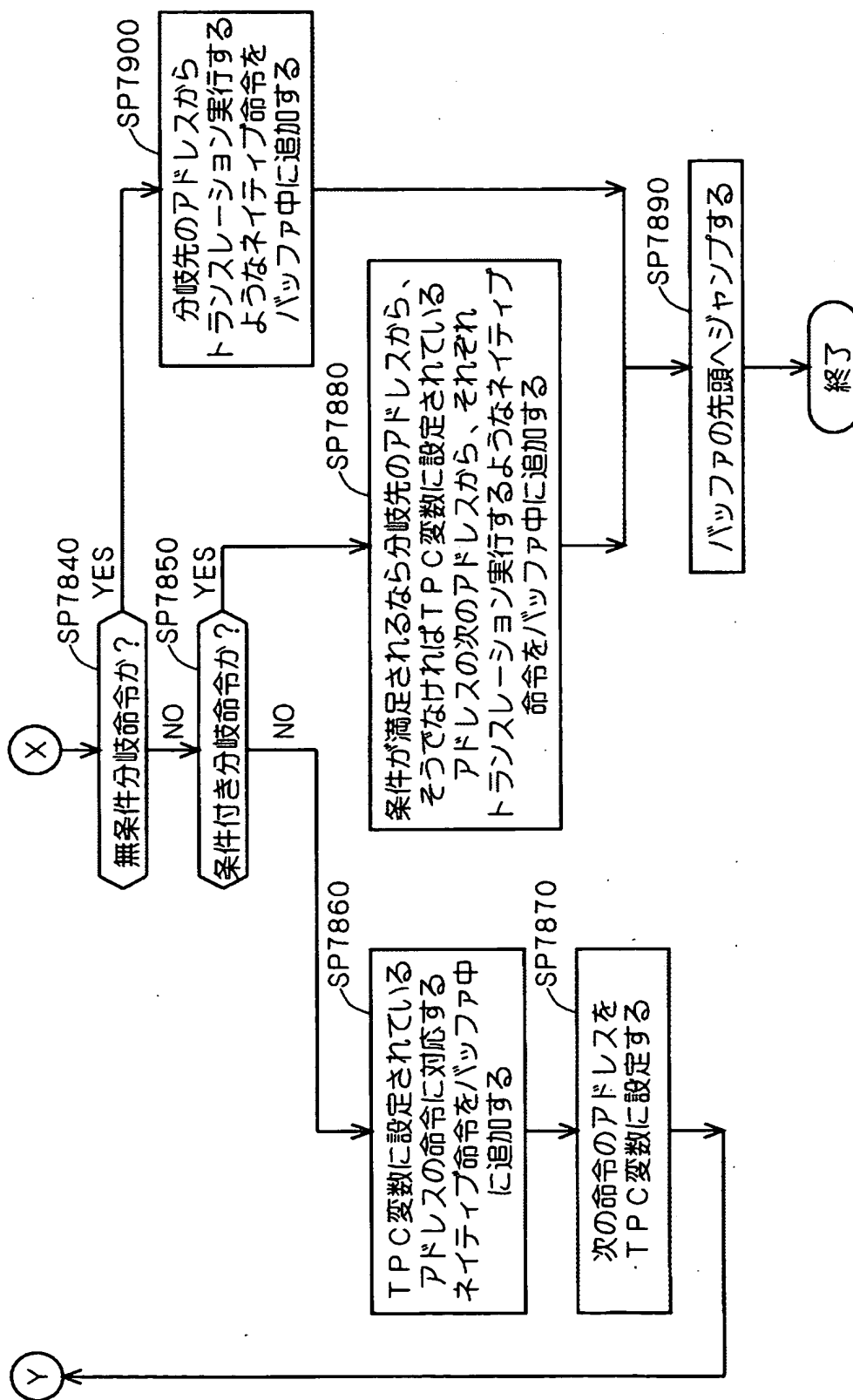
【図 2 0】



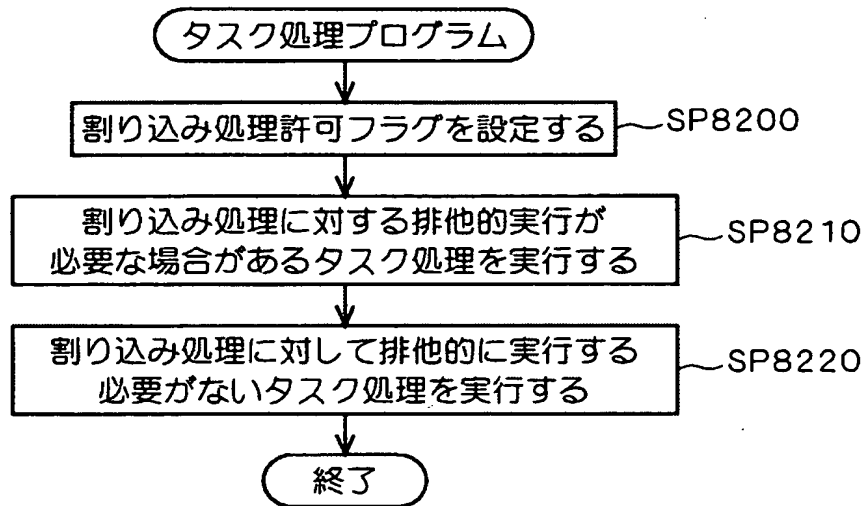
【図 21】



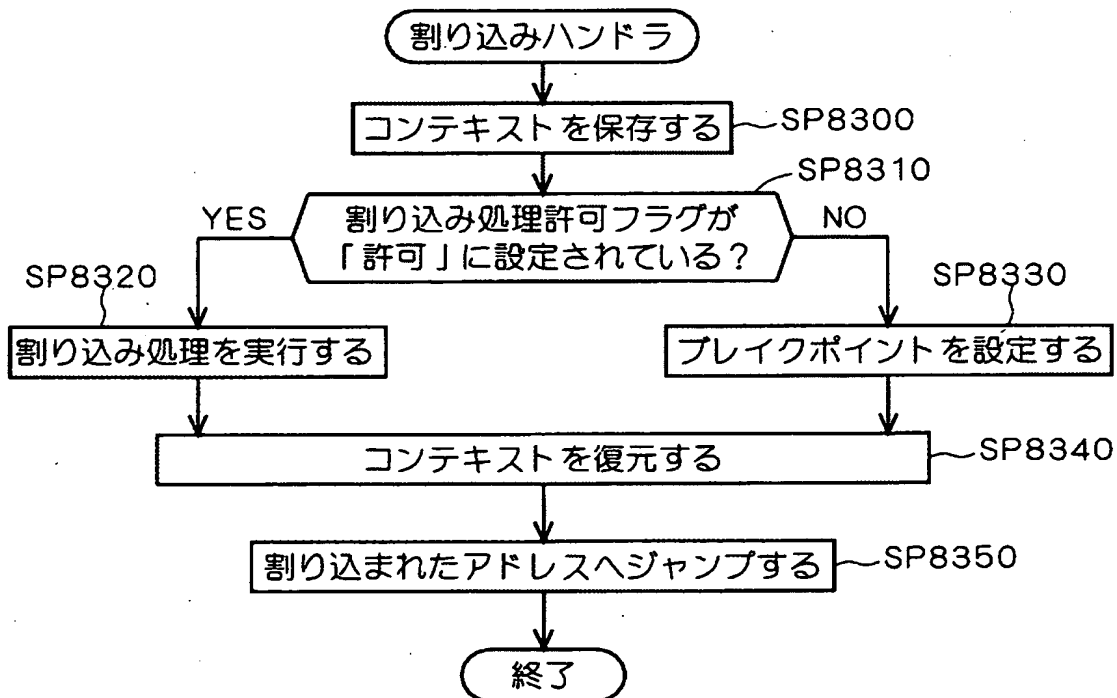
【図 22】



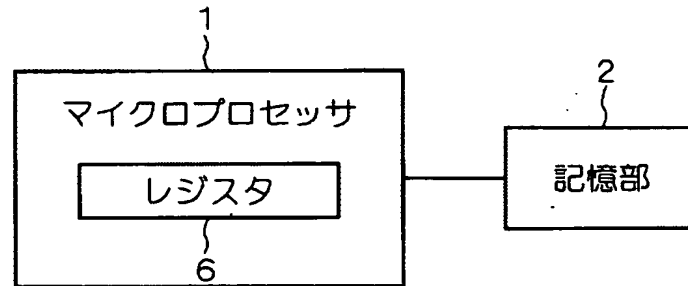
【図 2 3】



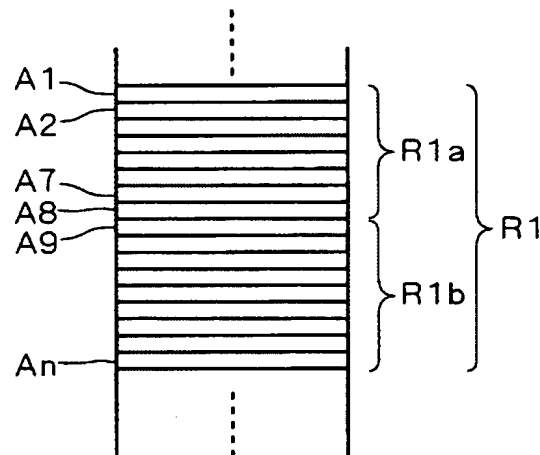
【図 2 4】



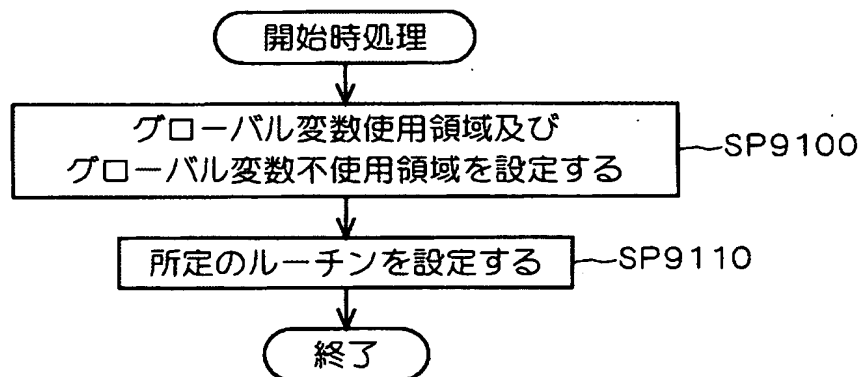
【図 25】



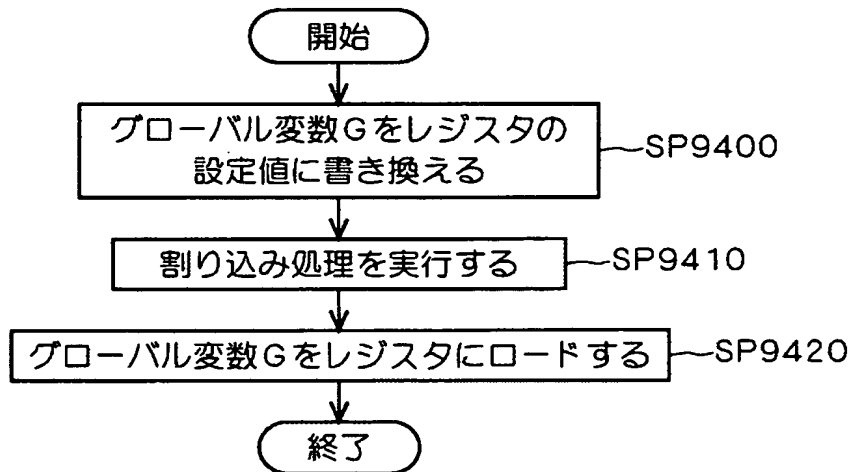
【図 26】



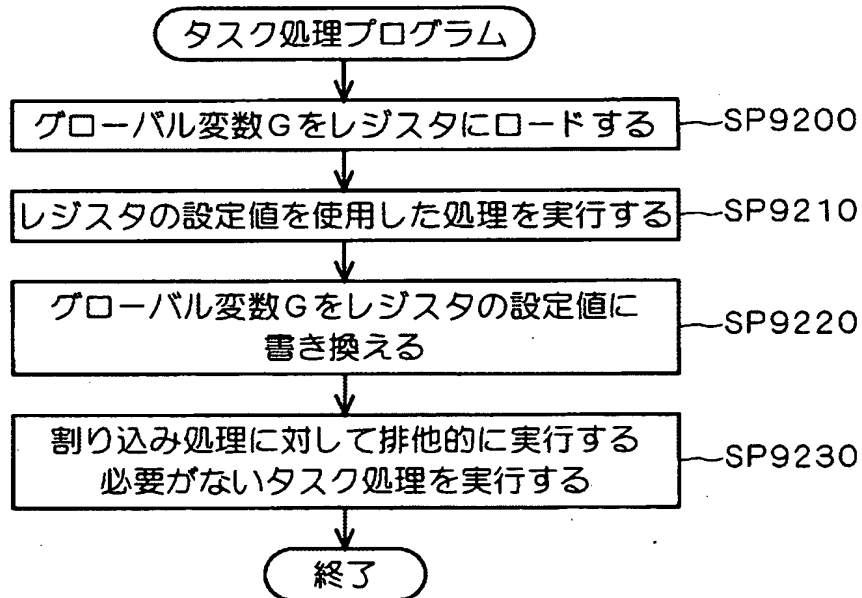
【図 27】



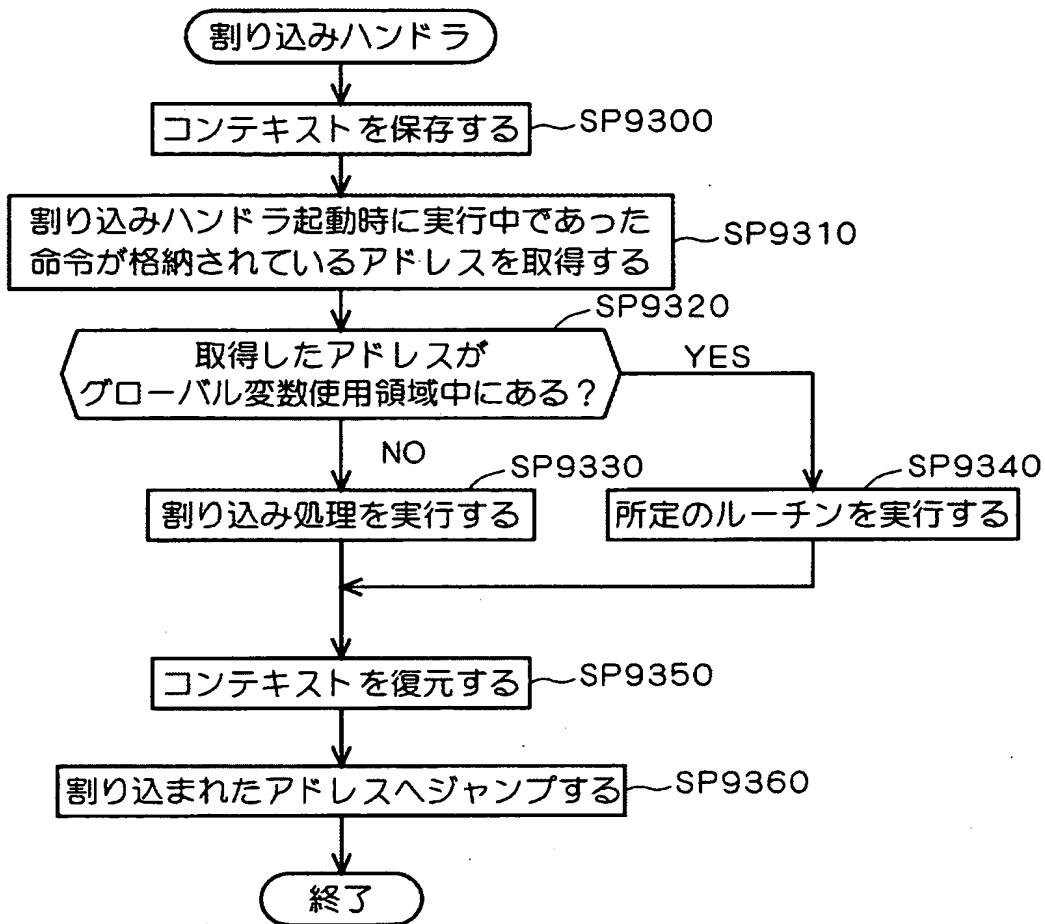
【図 2 8】



【図 2 9】



【図 3 0】



【書類名】 要約書

【要約】

【課題】 タスク処理の実行速度が低下することを回避しつつ割り込み処理を実行することが可能な割り込み制御方法を得る。

【解決手段】 タスク処理プログラムを実行している最中に割り込み要求が発生すると、タスク処理プログラムが中断されて、割り込みハンドラの実行が開始される。割り込みハンドラによって、割り込み処理許可領域 R 2 内に複数のブレイクポイントが設定される。再開されたタスク処理プログラムの実行は、複数のブレイクポイントのいずれかに、やがて到達する。すると、マイクロプロセッサ 1 は、タスク処理プログラムの実行を中断し、ブレイクポイントハンドラの実行を開始する。ブレイクポイントハンドラによって、割り込み処理が実行され、その後、ブレイクポイントの設定が解除される。

【選択図】 図 5

出 願 人 履 歴 情 報

識別番号 [000006013]

1. 変更年月日	1990年 8月24日
[変更理由]	新規登録
住 所	東京都千代田区丸の内2丁目2番3号
氏 名	三菱電機株式会社